Model order reduction for large-scale data assimilation problems

Green, Daniel

Award date:
2019

Awarding institution:
University of Bath

Model order reduction for large-scale data assimilation problems

submitted by

Daniel Luke Hoskins Green

for the degree of Doctor of Philosophy

of the

University of Bath

Department of Mathematical Sciences

May 2019

COPYRIGHT

Declaration of any previous submission of the work

The material presented here for examination for the award of a higher degree by research has not been incorporated into a submission for another degree.

.....

Daniel L.H. Green

Declaration of authorship

I am the author of this thesis, and the work described therein was carried out by myself personally. Chapters 3 to 5 were carried out in collaboration with my supervisor, Melina Freitag.

.....

Daniel L. H. Green

ABSTRACT

Data assimilation is an important method for incorporating data (typically observations) into a model. In this thesis we consider methods to reduce the size of the state space within the data assimilation process, focusing on the weak constraint four dimensional variational data assimilation approach (4D-Var).

The linearised system arising within the minimisation process can be formulated as a saddle point problem. A disadvantage of this formulation is the large storage requirements involved in the linear system. We present a low-rank approach which exploits the structure of the saddle point system using techniques and theory from solving large scale matrix equations to obtain an approximate solution which has significantly lower storage requirements. Three preconditioning approaches for the saddle point formulation of the data assimilation problem are applied to the iterative solving of the saddle point system using GMRES, and the low-rank method introduced in this thesis which introduces additional considerations.

In addition we present projection methods for reducing the dimension of the space the state of the system resides in in weak constraint 4D-Var. We apply the control theoretic balanced truncation model reduction method, and introduce randomised projection methods, sometimes known as sketching methods to the data assimilation setting.

Numerical experiments with the linear advection-diffusion equation, the shallow water equations and the nonlinear Lorenz-95 model demonstrate the effectiveness of applying these methods when compared to solving the data assimilation problem with standard approaches.

ACKNOWLEDGEMENTS

Acknowledging everyone who has assisted me over the course of my PhD is not an easy task. Arguably I could leave my place holder text 'everyone is great' and be done, but I am glad to be able to thank some of the many people who have helped make this thesis possible.

First and foremost I would like to thank my supervisor Melina Freitag for her guidance and extensive feedback over the course of this PhD. Despite the seemingly constant teaching load, she still managed to find the time to cover my work in helpful comments. Because of her and Alastair Spence introducing me to model reduction in my MSc project, I came to do a PhD, and for that I am incredibly thankful. I would like to thank Amos Lawless and Silvia Gazzola for being my examiners and reading this thesis, and EPSRC for providing the funding that allowed me to complete it, giving me the opportunity to attend and present at conferences nationally and internationally.

I have been incredibly fortunate to have had many friends who have ensured that I stayed sane during my PhD. My office, and the mathematics department as a whole have provided many sources of conversation, procrastination and fun over the past few years. Whether it was discussing the number of postboxes on campus, reading 'The Chronicles of Narnia', or planning a national conference, there has been no shortage of friendly faces to talk to. I am going to miss being a part of the department.

The 'Wednesday group' has been a great group of friends and we have had some amazing fun, I am very thankful to you all. Without Wednesday I would have seen a lot less weird films, eaten less cakes and not have met Bec.

Lastly, a big thank you goes to my family. Their support, encouragement, and to some extent competition has made this possible. You're the best.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 4: Preconditioning the data assimilation saddle point problem

Chapter 5: Projection methods for weak constraint variational data assimilation

LIST OF TABLES

Chapter 5: Projection methods for weak constraint variational data assimilation

CHAPTER 1

INTRODUCTION

In the modern world data is everywhere. This data arises from a whole host of different sources, and is used for a wide range of applications. The prevalence of computers, modern technology, and the internet has meant it is easier than ever to create, store, and interpret this data. The speed that data can be created often outstrips the computers which manipulate it.

Data assimilation is a method for using data in the form of observations to inform estimates. These observations (typically from a physical system) are combined with a numerical model of that physical system in order to create more accurate estimates of the actual state of the system. These estimates may be of the true state of the system, such as we consider in this thesis, or parameters involved in the model [120].

One example where data assimilation is used is numerical weather prediction, allowing meteorologists to update their predictions of the upcoming weather based on observations of the current temperature, pressure, humidity and many other properties [7, 66]. These observations are taken from a range of different sources and can be combined with the numerical models which exist for different parts of the atmosphere through data assimilation to obtain one collective forecast.

Data assimilation is commonly applied throughout the geosciences [22], in areas such as weather prediction [101, 102, 104], oceanography [59, 100, 138] and glaciology [25] to give some examples. With the greater interest in data across different industries and fields of science and technology, there are growing applications of data assimilation to these areas too e.g. [76, 139, 142], with further examples in [5].

Performing data assimilation can typically be an expensive process with the models used in the data assimilation method often arising from physical processes for many of these applications. The numerical models for these processes are often

computationally expensive to evaluate themselves. A further property which the traditional applications share is the vast dimensionality of the state vectors involved. In numerical weather prediction for example, the systems can have variables of order 10^8 and higher [82] to describe the current state of the atmosphere. Whilst the number of observations taken of the state is often also very large, there are typically significantly fewer observation points than the size of the state of the system by several orders of magnitude.

In this thesis we consider methods to reduce the size of the state space within the data assimilation process. In particular we focus on the weak constraint four dimensional variational data assimilation approach (weak constraint 4D-Var) and achieve this reduction in two different ways. Our first approach considers approximations to the vectors within the data assimilation process which when generated using existing approaches can have as many entries as 10^8. We propose an alternative low-rank solver for a saddle point system arising within the data assimilation problem, using techniques and theory from solving large scale matrix equations to obtain an approximate solution which has significantly lower storage requirements. The second approach draws inspiration from more traditional system theoretic model reduction methods. For this method we apply projection methods to the data assimilation problem hereby reducing the dimension of the space the state of the system resides in. We apply the control theoretic balanced truncation model reduction method, and randomised projection methods, sometimes known as sketching methods, to weak constraint 4D-Var. The resulting projected system is less computationally expensive and projecting back to the original space after obtaining a solution in the smaller dimensional space provides an approximate solution which is obtained in less computation time than working in the high dimensional space.

We use this chapter to introduce in greater detail some of the methods for applying data assimilation, before we set out the structure of this thesis.

1.1 | An introduction to data assimilation

There are two primary classes of data assimilation:

- *sequential* methods, where the assimilation of observations is performed at each timestep, and
- *variational* methods, where the assimilation is performed for all timesteps at once.

Prior to considering the application of these classes of methods, let us first describe the shared setting of data assimilation problems.

As stated above, the aim of data assimilation is to combine observations with a numerical model, in order to obtain a better estimate of the true state of the system. We consider the discrete-time nonlinear dynamical system

$$x_{k+1} = \mathcal{M}_k(x_k) + \eta_k, \tag{1.1}$$

where $x_k \in \mathbb{R}^n$ is the state of the system at time t_k and $\mathcal{M}_k : \mathbb{R}^n \to \mathbb{R}^n$ is the nonlinear model operator which evolves the state from time t_k to t_{k+1} for $k = 0, \dots N-1$. The terms $\eta_k \in \mathbb{R}^n$ represent the model error at time t_k, these are assumed to be Gaussian with zero mean and a known model error covariance matrix $Q_k \in \mathbb{R}^{n \times n}$.

Observations of this system, $y_k \in \mathbb{R}^{p_k}$ at time t_k for $k = 0, \dots N$ are obtained through an observation operator $\mathcal{H}_k : \mathbb{R}^n \to \mathbb{R}^{p_k}$:

$$y_k = \mathcal{H}_k(x_k) + \epsilon_k, \tag{1.2}$$

here $\epsilon_k \in \mathbb{R}^{p_k}$ is the observation error at t_k, these errors are also assumed to be Gaussian, with zero mean and an observation error covariance matrix $R_k \in \mathbb{R}^{p_k \times p_k}$. In general, the number of observations at each timestep satisfies $p_k \ll n$. The observation operator $\mathcal{H}_k$ may also be nonlinear, and have explicit time dependencies depending on the application area.

We assume that at the initial time we have an a-priori estimate of the state, which we refer to as the background state, and denote x_0^b. This is commonly the result of a short-range forecast, or a previous assimilation cycle, and is typically taken to be the first guess during the assimilation process. This background state has an error associated with it:

$$x_0 - x_0^b = e_0, \tag{1.3}$$

and we assume this error is also Gaussian with a zero mean and a background error covariance matrix $B \in \mathbb{R}^{n \times n}$.

We present now two approaches for data assimilation beginning with sequential methods in the next section.

1.2 | Sequential data assimilation

In sequential data assimilation methods, one assimilation is performed at each timestep in the assimilation window, and each is done one after the other. This is unlike the variational methods in Section 1.3 which we consider for the remainder of the thesis where all timesteps are assimilated at once.

The most commonly applied form of sequential data assimilation is Kalman filtering introduced in [75], and it remains one of the most popular approaches for data assimilation. There have been many modifications to the Kalman filter, some of which we consider in Chapter 2 when we present a literature review of existing approaches for applying model reduction methods, or utilising low-rank properties within data assimilation.

1.2.1 | Kalman filter

The Kalman filter [75] consists of two steps, a forecast step and an update or analysis step. We consider here the extended Kalman filter [58], an extension allowing for nonlinear model and observation operators $\mathcal{M}_k$ and $\mathcal{H}_k$ as in (1.1) and (1.2). This is achieved by generating the tangent linear model, and observation operators M_k and H_k by linearising $\mathcal{M}_k$ and $\mathcal{H}_k$ about x_k, and using these matrices for some elements of the Kalman filter. To initialise the Kalman filter we consider the background estimate to the state x_0^b, and the background error covariance matrix B as described in (1.3).

The first step of the Kalman filter is the *forecast step*. Here we evolve the model forward from our existing forecast to give an estimate of the current state:

$$x_k^f = \mathcal{M}_k(x_{k-1}^a), \tag{1.4}$$

where we take the background estimate to the state to be the initial forecast $x_0^a = x_0^b$.

In order to give a sense of the current error arising from the forecast, we update the predicted covariance matrix P^f using the (tangent linear) model operator M_k and model error covariance Q_k, giving

$$P_k^f = M_k P_{k-1}^a M_k^T + Q_k, \tag{1.5}$$

taking into account the evolution of the model, and the definition of the model error covariance. Here P_{k-1}^a is the corrected covariance matrix obtained in the analysis step. We take the initial error covariance estimate to be the background error

covariance matrix $P_0^a = B$.

In the *analysis step*, this forecast is corrected, using knowledge of the observations, and the covariances for the state and observation errors:

$$x_k^a = x_k^f + K_k(y_k - \mathcal{H}_k(x_k^f)), \tag{1.6}$$

$$P_k^a = (I - K_k H_k)P_k^f, \tag{1.7}$$

where K_k is known as the Kalman gain matrix, and is given by

$$K_k = P_k^f H_k^T (H_k P_k^f H_k^T + R_k)^{-1}. \tag{1.8}$$

This is continued for all timesteps and for each time t_k we obtain an estimate of the state x_k. When using linear model and observation operators and Gaussian errors, this estimate we obtain is the best linear unbiased estimate, however for the extended Kalman filter [58, 72], an approximation to the best linear unbiased estimate is obtained.

The computationally expensive steps when applying the Kalman filter are the inversion when generating the Kalman gain matrix (1.8), and propagating the error covariance P_k^f in (1.5), where at each timestep we must perform a matrix multiplication on the left by M_k and on the right by M_k^T. There have been a number of different methods proposed to alleviate these, some of which we present in Chapter 2 when we consider previous approaches to model reduction within data assimilation.

1.2.2 | Bayesian data assimilation

The Kalman filter assumes Gaussian distributions on the errors. The most general example of sequential data assimilation is Bayesian data assimilation which allows for different distributions. This approach determines a probability density function $\pi_k^a(x_k)$ at time t_k for the states x_k given the observations y_k. As with the Kalman filter this process consists of forecast and analysis steps.

When forecasting, the prior density $\pi_k^f(x_k)$ is calculated by propagating the analysis density $\pi_{k-1}^a(x_k)$ from time t_{k-1} to t_k using the model operator $\mathcal{M}_k$. As with the Kalman filter, the initial prior density is taken to be the probability density of the background estimate.

The analysis step updates this density, calculating the posterior density $\pi_k^a(x_k|y_k)$ using Bayes' formula:

$$\pi_k^a(x_k|y_k) = \frac{\pi_k^f(x_k)\pi(y_k|x_k)}{\pi(y_k)}.$$

Here the probability density of the observations $\pi(y_k)$ acts as a normalising constant, and the density of the data distribution $\pi(y_k|x_k)$ (sometimes referred to as the measurement model) is given by

$$\pi(y_k|x_k) = \phi(y_k - \mathcal{H}_k(x_k)),$$

using an error density ϕ for the observation errors.

This approach is the most general, allowing for non-Gaussian prior and posterior distributions however it is generally very expensive.

It can be shown that when considering linear operators and Gaussian probability densities, the Bayesian approach is equivalent to the Kalman filter and variational data assimilation [57, 81].

The following section details the variational data assimilation approach which we use for the new methodology we introduce in this thesis.

1.3 | Variational data assimilation

Variational data assimilation, initially proposed in [114, 115] finds its roots in optimisation, and is the other primary class of data assimilation methods. In variational data assimilation, all timesteps in the assimilation window are considered at once, in contrast to sequential methods where each timestep is assimilated one step at a time.

1.3.1 | Four dimensional variational data assimilation (4D-Var)

Four dimensional variational data assimilation (4D-Var) is so called for three spatial dimensions, plus time, and to differentiate it from three-dimensional variational data assimilation (3D-Var), where we do not consider multiple observation times. In 4D-Var, we find an initial state which minimises both the weighted least squares distance to the background state x_0^b (typically obtained from the previous forecast) and the observations y_k for an assimilation window $[t_0, t_N]$. We can consider 3D-Var as a special case of 4D-Var where the assimilation window consists of just one timestep.

Strong constraint 4D-Var

In strong constraint 4D-Var we assume that the model $\mathcal{M}_k$ in (1.1) is perfect, and apply this as a strong constraint within the minimisation process. This methodology was used in [115] where variational data assimilation was introduced. Hence when we find our minimising initial state, we minimise the cost function

$$\begin{aligned} J(x_0) &= \underbrace{\frac{1}{2}(x_0 - x_0^b)^T B^{-1}(x_0 - x_0^b)}_{J_b} + \underbrace{\frac{1}{2}\sum_{k=0}^{N}(y_k - \mathcal{H}_k(x_k))^T R_k^{-1}(y_k - \mathcal{H}_k(x_k))}_{J_o}, \\ &= \frac{1}{2}\|x_0 - x_0^b\|^2_{B^{-1}} + \frac{1}{2}\sum_{k=0}^{N}\|y_k - \mathcal{H}_k(x_k)\|^2_{R_k^{-1}}, \end{aligned} \tag{1.9}$$

subject to the *strong constraint* that $x_{k+1} = \mathcal{M}_k(x_k)$.

This cost function consists of two parts, J_b the background term, which penalises the background error arising from the a priori estimate of the state, weighted by the background error covariance B and J_o which penalises the observations y_k at all timesteps in the assimilation window, these are weighted accordingly by the observation error covariance matrices R_k. As in (1.2) and (1.3), we assume that the observation and background errors are Gaussian. The assumption of a Gaussian distribution allows us to define the errors by their mean and covariances. We assume that the background and observation errors have zero mean and covariances B and R_k respectively. The strong constraint 4D-Var problem is typically solved using the adjoint method [42, 126].

Weak constraint 4D-Var

The weak constraint formulation of 4D-Var arises from assuming an imperfect model as in (1.1), with $x_{k+1} = \mathcal{M}_k(x_k)+\eta_k$ where η_k denotes the model error. It is assumed that η_k is Gaussian with zero mean and covariance Q_k. The relaxation of the strong constraint $x_{k+1} = \mathcal{M}_k(x_k)$ is commonly used in sequential data assimilation as seen in Section 1.2.1, where the covariance matrix Q_k is used to update the predicted covariance in (1.5). For variational data assimilation, applying a weak constraint was also proposed in [115], however due to the computational cost was not commonly used until far more recently. In the past couple of decades however, there has been greater interest in weak constraint variational data assimilation, see for example [53, 55, 63, 90, 131, 133, 145].

In weak constraint 4D-Var we wish to find a state which minimises the weighted

least squares distance to the background state x_0^b, and the observations y_k, but additionally the weighted least squares distance between the model trajectory of this initial state x_k over the assimilation window $[t_0, t_N]$

Mathematically, we can write this as the minimisation of a cost function,

$$\begin{aligned} J(x) &= \underbrace{\frac{1}{2}(x_0 - x_0^b)^T B^{-1}(x_0 - x_0^b)}_{J_b} + \underbrace{\frac{1}{2}\sum_{k=0}^{N}(y_k - \mathcal{H}_k(x_k))^T R_k^{-1}(y_k - \mathcal{H}_k(x_k))}_{J_o} \\ &\quad + \underbrace{\frac{1}{2}\sum_{k=1}^{N}(x_k - \mathcal{M}_k(x_{k-1}))^T Q_k^{-1}(x_k - \mathcal{M}_k(x_{k-1}))}_{J_q}, \\ &= \frac{1}{2}\|x_0 - x_0^b\|^2_{B^{-1}} + \frac{1}{2}\sum_{k=0}^{N}\|y_k - \mathcal{H}_k(x_k)\|^2_{R_k^{-1}} + \frac{1}{2}\sum_{k=1}^{N}\|x_k - \mathcal{M}_k(x_{k-1})\|^2_{Q_k^{-1}}, \end{aligned} \tag{1.10}$$

where $x = [x_0^T, x_1^T, \ldots, x_N^T]^T$, and x_k is the model state at each timestep t_k for $k = 0, \ldots, N$. This formulation is known as the "state formulation" of weak constraint 4D-Var. We note that here we minimise over x as opposed to x_0 in strong constraint 4D-Var to account for the addition of model error. As a result we introduce the J_q term which penalises the model errors η_k over all timesteps, weighted by the corresponding model error covariances Q_k, thus incorporating the constraint $x_{k+1} = \mathcal{M}_k(x_k) + \eta_k$ into the objective function.

An equivalent approach, referred to as the "forcing formulation" [133] is to consider the minimisation in terms of the initial condition x_0 and the model errors η_k, which results in the cost function:

$$\begin{aligned} J(p) &= \frac{1}{2}(x_0 - x_0^b)^T B^{-1}(x_0 - x_0^b) + \frac{1}{2}\sum_{k=0}^{N}(y_k - \mathcal{H}_k(x_k))^T R_k^{-1}(y_k - \mathcal{H}_k(x_k)) \\ &\quad + \frac{1}{2}\sum_{k=1}^{N}\eta_k^T Q_k^{-1}\eta_k, \end{aligned} \tag{1.11}$$

subject to the constraint $x_{k+1} = \mathcal{M}_k(x_k) + \eta_k$, where $p = [x_0^T, \eta_1^T, \ldots, \eta_N^T]^T$.

The additional cost of weak constraint 4D-Var, and the difficulties in computing Q_k mean that it is not widely implemented in real world systems. However, accounting for this model error (with suitable covariances) leads to improved accuracy, and the added potential of longer assimilation windows [52, 53]. The saddle point formulation of (1.10) as used in [51, 53, 55] and Chapter 3 has also seen interest for the parallelisable nature of the matrix-vector products involved, we refer to [51] for

discussion on the parallelisation of this problem.

It is weak constraint 4D-Var which we consider for the majority of this thesis, in particular the state formulation (1.10).

1.3.2 | Incremental 4D-Var

To implement 4D-Var operationally, an incremental approach [35] is used. This is essentially the Gauss-Newton method, an iterative approach to solving nonlinear least squares problems, and applying it to the strong constraint 4D-Var problem generates an approximation to the solution of $x_0 = \text{argmin } J(x_0)$, extending naturally to weak constraint 4D-Var. (See Chapter 3 for details).

Iterates ℓ are introduced such that

$$x_0^{(\ell+1)} = x_0^{(\ell)} + \delta x_0^{(\ell)}, \tag{1.12}$$

where $x_0^{(\ell)}$ denotes the ℓ-th approximation to x_0, with the initial state $x_0^{(0)}$ taken to be the background estimate x_0^b, with $\delta x_0^{(0)} = 0$.

Linearising the cost function (1.9) around the model trajectory forecast from the estimate $x_0^{(\ell)}$, we can write it in terms of the increment $\delta x_0^{(\ell)}$:

$$\tilde{J}(\delta x_0^{(\ell)}) = \frac{1}{2}\|\delta x_0^{(\ell)} - b_0^{(\ell)}\|^2_{B^{-1}} + \frac{1}{2}\sum_{k=0}^{N} \|d_k^{(\ell)} - H_k \delta x_k^{(\ell)}\|^2_{R_k^{-1}}. \tag{1.13}$$

Here $M_k \in \mathbb{R}^{n\times n}$ and $H_k \in \mathbb{R}^{p_k \times n}$, are linearisations of $\mathcal{M}_k$ and $\mathcal{H}_k$ about the current state trajectory $x^{(\ell)}$ obtained from evolving $x_0^{(\ell)}$ forward. The increment δx_k satisfies the linear dynamical equation

$$\delta x_{k+1} = M_k \delta x_k.$$

Furthermore, we introduce the vectors $b_0^{(\ell)}$ and $d_k^{(\ell)}$:

$$b_0^{(\ell)} = x_0^b - x_0^{(\ell)}, \tag{1.14}$$

$$d_k^{(\ell)} = y_k - \mathcal{H}_k(x_k^{(\ell)}), \tag{1.15}$$

the vectors $d_k^{(\ell)}$ are referred to as the *innovation* vectors in some applications.

Minimising the cost function (1.13) is known as the *inner loop*, whilst the update of the model trajectory $x_i^{(\ell)}$ is the *outer loop*. The minimisation can be performed using an iterative method, or through solving the gradient equation at the minimum ($\nabla \tilde{J} = 0$). This minimisation yields a new increment $\delta x_0^{(\ell)}$ from which we can update

the current estimate for $x_0^{(\ell)}$ using (1.12).

This method can be extended naturally to the weak constraint setting for 4D-Var [131], and is detailed further in Chapter 3.

1.3.3 | Connections between the approaches

Under the assumption of linear model and observation operators, with background and observations errors arising from a Gaussian distribution, it can be shown that the Kalman filter and strong constraint 4D-Var are equivalent to Tikhonov regularisation [57]. Another possibility for considering the data assimilation problem is to consider probability density functions for the state given observation data. This leads to the Bayesian approach to data assimilation, where posterior densities for the state at time t_k are calculated. In [57, 81] it is shown that the Bayesian approach is equivalent to the Kalman filter and strong constraint 4D-Var when considering linear operators and Gaussian probability densities. The Bayesian approach is the most general, allowing for non-Gaussian prior and posterior distributions however it is generally very expensive.

There has been considerable investigation into how sequential and variational methods relate and compare when extensions, hybrid methods and low-rank approaches are considered see for example [6, 28, 29, 41, 52, 57] and the references therein. In the case of weak constraint 4D-Var, it has been shown that it is equivalent to Kalman smoothing [52]. Whilst Kalman filtering assimilates observations as they become available, using past and present observations to predict the state, Kalman smoothing [2, 32] aims to estimate the state of the system using past, present and possibly future observations. Let us assume that t_K is the current time, with $1 \leq k \leq K$. As described in Section 1.2.1, the Kalman filter estimates the state x_k using observations $y_1, \ldots, y_k$. In contrast the Kalman smoother allows the estimation of the state x_k using observations $y_1, \ldots, y_K$. More generally, the Kalman smoother can estimate all states $x_1, \ldots x_K$ using the observations $y_1, \ldots, y_K$, which is precisely the aim in weak constraint 4D-Var.

The focus of this thesis is on reduced order solutions to the weak constraint four dimensional variational data assimilation problem. As a result, considering how the methods introduced here would translate to other approaches is beyond the scope of this work.

1.4 | Structure of the thesis

This thesis is arranged as follows. In the following chapter, Chapter 2 we present a review of the existing methodology for applying some form of model order reduction to the data assimilation problem. These approaches include applying traditional model reduction methods such as the control theoretic balanced truncation method, and nonlinear model reduction methods such as proper orthogonal decomposition to the data assimilation problem. There have been further methods employing low-rank approximations of covariance matrices to reduce the complexity of the computations within the Kalman filter and variational data assimilation. This review allows us to place the new methods introduced in this thesis in the wider context of the existing literature.

In Chapter 3 we introduce the saddle point formulation for the weak constraint 4D-Var problem, and consider the relationship between the resulting saddle point matrix written in terms of Kronecker products and matrix equations. The main contribution in this chapter is in Section 3.2 where we propose an approach to solve the saddle point problem exploiting this structure using techniques and theory from solving large scale matrix equations, allowing us to obtain a low-rank solution. We present a new low-rank form of inexact GMRES (LR-GMRES) which returns low-rank solutions requiring considerably less storage than standard GMRES. After proving the existence of such solutions in Section 3.2.2, this chapter is concluded by presenting numerical experiments comparing this new low-rank solver with GMRES. We examine three example problems displaying different characteristics, the one dimensional advection-diffusion equation, a linearised two-dimensional shallow water equations example and the nonlinear and chaotic Lorenz system which requires a small extension to the initially presented method. We observe that the low-rank approach introduced here is successful using both linear and nonlinear models, achieving close approximations to the full-rank solutions with storage requirements as low as 1% of those needed by the full-rank approach, which can be obtained in less time than through GMRES.

Chapter 4 presents three preconditioning approaches for the saddle point formulation of the data assimilation problem. These preconditioners are applied to the solution of the saddle point system, and the low-rank method introduced in Chapter 3. In Section 4.2 we introduce approximations to the matrices in the saddle point problem which must be considered when constructing preconditioners, and investigate the effect of these approximations and preconditioners on the spectra of the saddle point system. Truncating the inverse of a matrix in Kronecker form,

we investigate further approximations which provide faster convergence but require a greater number of matrix vector products at each iteration. We apply the three preconditioners with these approximations to the GMRES and LR-GMRES methods for the three example problems introduced in the previous chapter and consider the efficacy of these preconditioning approaches. The low-rank method introduces additional considerations for preconditioners which must be taken into account, and we observe that the method acts in some sense like a projected preconditioner itself, with preconditioners being less effective than for GMRES.

In Chapter 5 we consider the application of projection methods to the data assimilation problem, hereby reducing the dimension of the state space. After setting the problem for a general projection, we consider two approaches. Firstly we extend previous work applying the control theoretic balanced truncation method to the strong constraint 4D-Var problem to the weak constraint scenario, introducing the necessary concepts from control theory. Furthermore we introduce randomised projection methods, sometimes known as sketching methods, to the data assimilation problem. The approximation error obtained by solving the projected problem rather than the full-size problem is considered in Section 5.5. We finish this chapter with numerical experiments comparing these projection methods to solving the full-sized system, using the example systems from previous chapters. A further consideration which must be made is the variability of the randomised projections, and this is also addressed in Section 5.6. We observe that projection methods result in close levels of error to those obtained using the full scale minimisation, despite the reduced space being significantly smaller.

The thesis concludes with a summary of the results obtained in Chapter 6, and some outlooks for future research.

CHAPTER 2

MODEL REDUCTION APPROACHES FOR DATA ASSIMILATION

In the past decade or two in particular, model reduction methods have started being used in a number of different applications. Data assimilation is no exception, and there have been a number of papers applying model reduction techniques and ideas to both sequential and variational data assimilation. There are a large number of model order reduction techniques and approaches which have been considered in the data assimilation setting, for many different applications. As listed in Chapter 1, there are several approaches for applying data assimilation, and as a result this literature review may be incomplete. In this chapter we detail some of these existing approaches which have been considered.

One of the difficulties in applying traditional model reduction techniques from control theory within data assimilation is that in many applications for data assimilation, the model operator is nonlinear and time-dependent. Some of the more popular system theoretic approaches for model reduction such as balanced truncation and IRKA (iterative rational Krylov algorithm) generally work only for linear (and stable) models, necessitating linearisation of the model. Due to the linearisation within incremental 4D-Var this can be accounted for, though these methods typically require additionally that the system is time-invariant.

This requirement may not be too restrictive in applications depending on the number of time-steps the assimilation is performed over as it may be a reasonable assumption that the model does not vary for some range of time.

Alternative methods suited to nonlinear and time dependent systems such as POD (proper orthogonal decomposition) and POD-DEIM (discrete empirical interpolation method) [31] can be applied to generate reduced order systems. Further-

more there have been extensions to balanced truncation such as [80, 113, 116] to allow for time varying systems.

When these model reduction approaches are applied in other settings, the offline cost of producing a reduced model is amortised by reusing the same reduced system over multiple runnings. However in data assimilation, each assimilation cycle (typically) leads to a new system which must be then reduced. Hence the cost is freshly incurred each time, unless a linear time-invariant system is considered. As such other approaches have been applied which do not consider the model and instead investigate low-rank covariance matrices, or sampling approaches. An alternative method is to consider low-rank solution techniques within the minimisation process, such as we consider in Chapter 3 (and the paper [55]).

In this chapter we review methods applied to the two families of data assimilation methods we introduced in Chapter 1: sequential methods with a focus on Kalman filters and variational data assimilation methods which are the focus on the remainder of the thesis. Let us first consider the reduced order modifications made to Kalman filters.

2.1 | Kalman filters

Of the sequential data assimilation methods, the approach which is most frequently taken is the Kalman filter, which we introduced in Section 1.2.1. There has been investigation into low-rank implementations of the Kalman filter, with reduced rank filters such as reduced rank square root filters in [137] and the singular evolutive extended Kalman (SEEK) filter [100], considering an ensemble as in [48] or combining these ideas [129].

In this section we present only a summary of some of these approaches. For greater detail, we refer the reader to [5] and the references therein.

2.1.1 | Reduced rank filters

Reduced rank filters present a method for overcoming one of the computationally expensive parts in the Kalman filter, the propagation of the error covariance in (1.5):

$$P_k^f = M_k P_{k-1}^a M_k^T + Q_k.$$

The reduced rank filters were introduced in [33, 137], and work with low-rank covariance matrices, thereby reducing the computational cost. The following method

is one of the best known reduced rank square root (RRSQRT) filters, the singular evolutive extended Kalman (SEEK) filter introduced in [100] and supposes that the error covariance P_k^a in (1.7) can be approximated by

$$P_k^a \approx S_k^a (S_k^a)^T,$$

for all k, where $S_k^a \in \mathbb{R}^{n\times r}$ with $r \ll p, n$ is a low-rank matrix.

This assumption allows (1.5) to be rewritten as

$$P_k^f = M_k S_{k-1}^a (S_{k-1}^a)^T M_k^T + Q_k, \tag{2.1}$$

or indeed

$$P_k^f = \tilde{S}_k^f (\tilde{S}_k^f)^T + Q_k, \quad \text{where} \quad \tilde{S}_k^f = M_k S_{k-1}^a. \tag{2.2}$$

If additional restrictions are made on the rank of Q_k, [5, 26], such that the rank of P_k^f is the same as that of P_k^a, it can be assumed further that $P_k^f = S_k^f (S_k^f)^T$. Hence the Kalman gain matrix can be replaced with a lower cost one:

$$K_k = S_k^f (I_r + (H_k S_k^f)^T R_k^{-1} (H_k S_k^f))^{-1} (H_k S_k^f)^T R_k^{-1}. \tag{2.3}$$

As such the resulting corrected forecast is as before:

$$x_k^a = x_k^f + K_k(y_k - \mathcal{H}_k(x_k^f)),$$

with the corrected covariance matrix:

$$P_k^a = S_k^a S_k^a, \quad \text{where } S_k^a = S_k^f (I_r + (H_k S_k^f)^T R_k^{-1} (H_k S_k^f))^{-\frac{1}{2}}. \tag{2.4}$$

We observe that the matrix inversion required for the inverse of the square root in (2.4) and in the Kalman gain matrix in (2.3) is an $r \times r$ matrix in contrast to the $p \times p$ matrix being inverted in (1.8). This results in a method which reduces the complexity of the Kalman filter. A consideration to be made is the initial choice of covariance matrices, and requires a low-rank approximation to be made to take as the initial choice $P_0^f = B \approx S_0^f (S_0^f)^T$. Investigations into this include [106], which also has applications in variational data assimilation. Further extensions have been made to the SEEK filter to allow for nonlinear models see for example [138].

2.1.2 | Ensemble Kalman filters

An alternative approach for reducing the complexity of the Kalman filter by not requiring the covariance matrices to be explicitly formed is the ensemble Kalman filter (EnKF) proposed in [48]. To apply this method a collection (ensemble) of m state vectors $x_{k(i)}$, $i = 1, \dots m$ at timestep k are formed, where it is assumed that the number of ensemble members $m \ll n$, the dimension of the state. The ensemble are themselves updated and propagated, and the variability of the states leads to an estimate for the covariance of the error. With the exception of computing the Kalman gain matrix, the operations performed using the ensemble members are independent, and hence the EnKF can easily be parallelised.

Each ensemble member $x_{k(i)}$ is evolved forward using (1.1), adding noise η_k with zero mean and covariance Q_k:

$$x^f_{k(i)} = \mathcal{M}_k(x^a_{k-1(i)}) + \eta_k. \tag{2.5}$$

The covariance matrices are obtained by Monte Carlo estimators, with the forecast error covariance matrix computed as

$$P^f_k = \frac{1}{m-1}\sum_{i=1}^{m}(x^f_{k(i)} - \bar{x}^f_k)(x^f_{k(i)} - \bar{x}^f_k)^T, \quad \text{with} \quad \bar{x}^f_k = \frac{1}{m}\sum_{i=1}^{m} x^f_{k(i)}. \tag{2.6}$$

There are different approaches for the analysis step of ensemble Kalman filters, which can be generalised to two categories, stochastic approaches such as the perturbed observation EnKF [48] and deterministic approaches such as Ensemble Square Root Filters [129] akin to those in Section 2.1.1.

We first consider the perturbed observation Kalman filter. Here the ensemble members are updated in the analysis step using perturbed observations

$$x^a_{k(i)} = x^f_{k(i)} + K_k\left[y_k + \epsilon_{y_k} - \mathcal{H}_k(x^f_{k(i)})\right], \qquad i = 1, \dots, m, \tag{2.7}$$

where ϵ_{y_k} is drawn from a Gaussian distribution with zero mean and covariance R_k. Here the Kalman gain matrix is as in (1.8)

$$K_k = P^f_k H^T_k \left(\frac{1}{m-1}\sum_{i=1}^{m}\left[\mathcal{H}_k(x^f_{k(i)} - \bar{x}^f_k)\right]\left[\mathcal{H}_k(x^f_{k(i)} - \bar{x}^f_k)\right]^T + R_k\right)^{-1}, \tag{2.8}$$

using the Monte Carlo estimate P^f_k (2.6). The corrected covariance P^a_k can thus be

calculated as

$$P_k^a = \frac{1}{m-1}\sum_{i=1}^{m}(x_{k(i)}^a - \bar{x}_k^a)(x_{k(i)}^a - \bar{x}_k^a)^T, \quad \text{with} \quad \bar{x}_k^a = \frac{1}{m}\sum_{i=1}^{m} x_{k(i)}^a, \tag{2.9}$$

though is not necessary to be computed for applying this method. The introduction of the random perturbations is to yield the same analysis error covariance matrix

$$P_k^a = (I - K_k\mathcal{H}_k)P_k^f,$$

as in the original formulation of the Kalman filter (1.7) when taking the expectation over the random noise [5].

An alternative approach which does not introduce additional noise through perturbation of the observations is to consider updating the ensemble simultaneously instead of updating each ensemble member individually. Similarly to Section 2.1.1 we observe that

$$P_k^f = X_k^f(X_k^f)^T, \tag{2.10}$$

where the columns of $X_k^f \in \mathbb{R}^{n\times m}$ are given by the normalised perturbations

$$[X_k^f]_i = \frac{x_{k(i)}^f - \bar{x}^f}{\sqrt{m-1}}.$$

Transforming the forecast ensemble to the observation space results in

$$y_{k(i)}^f = \mathcal{H}_k(x_{k(i)}^f), \tag{2.11}$$

and hence computing the mean and perturbations we obtain

$$\bar{y}^f = \frac{1}{m}\sum_{i=1}^{m} y_{k(i)}^f, \qquad [Y_k^f]_i = \frac{y_{k(i)}^f - \bar{y}^f}{\sqrt{m-1}}. \tag{2.12}$$

where $[Y_k^f]_i$ is the i-th column of Y_k^f.

The Kalman gain matrix can then be written as

$$K_k = X_k^f(Y_k^f)^T(Y_k^f(Y_k^f)^T + R_k)^{-1}. \tag{2.13}$$

We note that a similar approach can be taken to this for the perturbed observation EnKF, with a Y_k^f including the perturbation [5, 48].

From here we can update the ensemble mean and perturbation

$$\bar{x}_k^a = \bar{x}_k^f + K_k(y_k + \bar{y}_k^f), \tag{2.14}$$
$$X_k^a = X_k^f T, \tag{2.15}$$

where the matrix T is chosen such that

$$P_k^a = X_k^a (X_k^a)^T = X_k^f T (X_k^f T)^T \tag{2.16}$$
$$\approx (I - K_k H_k) P_k^f, \tag{2.17}$$

as in the original formulation of the Kalman filter (1.7). The matrix T is not uniquely defined by this and thus there have been multiple variants of the ensemble square root Kalman filter, see for example [3, 21, 129, 141].

Hybrid methods

There has been investigation in recent years into methods which combine the ideas of ensemble Kalman filters as described in this section and variational data assimilation methods. It is not typically the methods themselves which are combined, but the error covariances obtained from the methods. In variational methods a static predetermined background covariance matrix B is used, whilst methods such as the ensemble Kalman filter estimate the flow-dependent error covariance P_k^f during the assimilation process. These methods are referred to as ensemble variational (EnVar) hybrid methods, and have led to similar or improved performance over traditional EnKF or variational methods [68, 90, 91, 92], with extensions to weak constraint 4D-Var [41, 53]. A simple blending implementation for an EnVar approach is to replace the background covariance matrix B with the covariance matrix

$$C = \gamma B + (1 - \gamma) P^f, \tag{2.18}$$

where $\gamma \in [0, 1]$ is a scalar parameter which controls the blending of the covariances. The cost function and updating of the ensemble is dependent on the EnKF approach taken, with a stochastic method necessitating the inclusion of the observation permutations as above. This approach was first proposed in [68] for hybridising the EnKF and 3D-Var, but has since been extended to 4D-Var (see for example [28, 91]).

Ensemble approaches have been popular for data assimilation, with the parallelisability of the methods resulting in computational efficiency. The number of

ensemble members is typically taken to be significantly smaller than the size of the state, which naturally leads to a low-rank covariance matrix $P_k^f = X_k^f (X_k^f)^T$ resulting in further savings within the implementation.

2.1.3 | Balanced truncation within the Kalman filter

In a different approach to the above, a control theoretic technique can be applied. In [49], the balanced truncation model reduction method [96] is applied within the Kalman filter.

Here the linearised model and observation operators M_k and H_k are projected onto a lower dimensional space, with the error covariance and Kalman gain matrices being computed using these reduced operators and transformed back to the full space when updating the state estimate. The dimension of the reduced space is taken to be $r \ll n$ leading to significant reductions in the complexity of the Kalman filter. The reduced model and observation operators are defined as

$$\hat{M}_k = U^T M_k V \in \mathbb{R}^{r \times r},$$
$$\hat{H}_k = H_k V \in \mathbb{R}^{p_k \times r},$$

where the matrices U and V are obtained through balanced truncation. For further discussion on balanced truncation we refer to Chapter 5 and [4].

The reduced error covariance matrices $\hat{P}_k^f$ are predicted using the formula

$$\hat{P}_k^f = \hat{M}_k \hat{P}_{k-1}^a \hat{M}_k^T + \hat{Q}_k, \tag{2.19}$$

where $\hat{Q}_k$ is the model error covariance projected onto the reduced space: $\hat{Q}_k = U^T Q_k U$. The correction to the error covariance is then

$$\hat{P}_k^a = (I_r - \hat{K}_k \hat{H}_k) \hat{P}_k^f, \tag{2.20}$$

with the reduced order Kalman gain matrix $\hat{K}_k$ given by

$$\hat{K}_k = \hat{P}_k^f \hat{H}_k^T (\hat{H}_k \hat{P}_k^f \hat{H}_k^T + R_k)^{-1}. \tag{2.21}$$

If $r \ll p$, the inversion in (2.21) can be computed in a cheaper way by applying the Sherman-Morrison formula. When used to update the state, the reduced order

Kalman gain matrix $\hat{K}_k$ must be projected back to the original dimension

$$x_k^a = x_k^f + V\hat{K}_k(y_k - \mathcal{H}_k(x_k^f)).$$

Here the complexity of the Kalman filter is reduced by projecting the model and observation operators onto a lower dimensional space, and generating covariance and Kalman gain matrices of a smaller dimension. In [49] it is assumed that the time dependent system underlying the problem has a time-invariant dominant part on which balanced truncation is performed. This allows the projection matrices generated by balanced truncation to be used over multiple timesteps, and amortises the cost of performing balanced truncation.

2.2 | Variational data assimilation

Variational data assimilation has seen less specific development of low-rank or reduced order methods in contrast to sequential data assimilation. In [71] it is suggested that in order to reduce the computational costs involved in the minimisation of incremental 4D-Var, a linear simplification operator such as a projection can be used. It is the specification of this simplification operator which determines the efficacy of the reduced order method.

Simplified or reduced order models are implemented within the operationally used incremental 4D-Var. As introduced in Section 1.3.2, in incremental 4D-Var we consider an increment δx and solve a linearised cost function within an inner loop. The model matrices used in this inner loop may be approximations which are cheaper to compute or apply, or reduced order matrices lowering the complexity of the method.

Over the years different approaches have been considered for simplifying the model matrices used within incremental 4D-Var. These simplified models may be obtained using a lower resolution model with fewer grid points or simplification of the physics [130], or through a model reduction method such as balanced truncation [23, 24, 84, 85, 96].

The earliest approaches considered a coarse grid and a lower resolution model, and pre-date the use of incremental 4D-Var, with investigation into how close the coarse resolution should be to the full resolution used in the forecast to retain a level of accuracy. We refer to [127, 130] and the literature compilation of [34] for these approaches. Development of incremental methods which allow for multiple resolutions over different inner loops was considered in [136] and implemented operationally for

some numerical weather prediction applications. However this approach has since been shown to have convergence problems in contrast to the standard incremental 4D-Var method [132] in higher dimensions. Alternative multi-level approaches have been considered in recent years, such as using a multi-grid solver, or multi-level approximations within the incremental 4D-Var process [27, 40]. These approaches combine the accuracy of fine resolution grids, with the speed and reduced complexity of coarse grids.

Here we present a short summary of a selection of approaches, referring the reader to [5] and the references therein for greater detail.

2.2.1 | Reduced 4D-Var

There have been constructions for reducing the dimensionality of strong constraint 4D-Var by approximating the initial state x_0. In [45, 107] it is assumed that the initial state x_0 is contained in a space of reduced-dimension $r \ll n$ about the background estimate x_0^b:

$$x_0 = x_0^b + \sum_{i=1}^{r} c_i w_i,$$

where c_i are real coefficients, and the linearly independent vectors w_i contain the main directions of variability in the system.

When considering incremental 4D-Var, this results in

$$\delta x_0 = \sum_{i=1}^{r} c_i w_i, \tag{2.22}$$

where we have dropped the inner loop notation (ℓ) and thus the incremental cost function to be minimised in reduced 4D-Var (c.f. (1.13)) becomes

$$\begin{aligned} \tilde{J}_r(c_1, \ldots, c_r) = \frac{1}{2} \left\| \left(\sum_{i=1}^{r} c_i w_i \right) - b_0 \right\|^2_{B_r^{-1}} \\ + \frac{1}{2} \sum_{k=0}^{N} \left\| d_k - H_k M_k \cdots M_1 \left(\sum_{i=1}^{r} c_i w_i \right) \right\|^2_{R_k^{-1}}. \end{aligned} \tag{2.23}$$

Here B_r is the background error covariance in the reduced space, which approximates B in the full space through

$$B \approx W B_r W^T, \tag{2.24}$$

where the columns of W are the vectors w_i.

Minimisation of this reduced cost function takes place in a space of dimension $r \ll n$, leading to a significant saving in computational expense. The efficacy of the approach is dependent on the choice of vectors w_i, with a selection of different methods considered in [45]. This reduced method for incremental 4D-Var has been used to initialise a full dimension incremental 4D-Var in order to achieve faster convergence of the method, and computational savings [106].

A comparison between reduced 4D-Var and the SEEK filter was performed in [105], with both methods producing similar results. Hybrid methods as described in Section 2.1.2 using this formulation of incremental 4D-Var have been proposed in [78, 105] with improved accuracy.

2.2.2 | Proper Orthogonal Decomposition within 4D-Var

A similar and related approach for generating a reduced order model for use within 4D-Var is through proper orthogonal decomposition (POD), this procedure is also known as the Karhunen-Loève expansion, principal component analysis, or empirical orthogonal functions in different fields [123], and can be considered as an application of the singular value decomposition (SVD) to the approximation of general dynamical systems [4]. In [30, 37, 128] this method is applied by taking snapshots $x_{(i)}$, $i = 1, \ldots, m$ of the state evolution at various timesteps during the data assimilation window, where the number of snapshots is significantly less than the dimension of the state space ($m \ll n$). The mean $\bar{x}$ of this ensemble is taken, and a matrix of snapshots is computed from the perturbations from the mean:

$$\bar{x} = \frac{1}{m}\sum_{i=1}^{m} x_{(i)}, \qquad [X]_i = x_{(i)} - \bar{x}, \tag{2.25}$$

where $[X]_i$ denotes the i-th column of X.

Performing the singular value decomposition $X = U\Sigma V^T$ on this matrix of snapshots, allows a reduced order control to be obtained by projecting $x_0 - \bar{x}$ onto the POD space spanned by the left singular vectors:

$$x_0 - \bar{x} = U\eta = \sum_{i=1}^{m} \eta_i u_i. \tag{2.26}$$

The matrix of left singular vectors $U \in \mathbb{R}^{n\times m}$, is referred to in POD literature as the POD basis. The resulting minimisation problem is akin to (2.23), yielding the

optimal coefficients $\eta_1, \ldots, \eta_m$, with the approximate solution to the 4D-Var problem obtained through (2.26).

An alternative approach using POD to reduce the complexity in the 4D-Var process is to use the POD basis U as a projection matrix [123] in order to form a reduced order forward model. To apply this method we assume that $x_k \approx U\hat{x}_k$. The reduced order forward model is constructed using a Petrov-Galerkin projection, taking a matrix $W \in \mathbb{R}^{n\times m}$ such that $W^T U = I_m$.

The resulting model is applied as a constraint to the minimisation of a reduced order cost function:

$$\hat{J}_{POD}(\hat{x}_0) = \frac{1}{2}\|U\hat{x}_0 - x_0^b\|^2_{B^{-1}} + \frac{1}{2}\sum_{k=0}^{N}\|y_k - \mathcal{H}_k(U\hat{x}_k)\|^2_{R_k^{-1}}, \tag{2.27}$$

subject to the constraint of the reduced order model

$$\hat{x}_{k+1} = \hat{\mathcal{M}}_k(\hat{x}_k), \quad \hat{\mathcal{M}}_k(\hat{x}_k) = W^T\mathcal{M}_k(U\hat{x}_k). \tag{2.28}$$

This minimisation takes place in a lower dimensional space of size $m \ll n$ than the full space formulation of strong constraint 4D-Var (1.9), leading to a reduction in the complexity of the method. Using the POD basis as a projection could also be used within the incremental 4D-Var inner loop as a simplification operator as in [71].

Different implementations of these POD methods have been proposed with variations to the generation of the POD basis U, through standard and tensorial POD methods and the POD-DEIM (discrete empirical interpolation method) [31] approach [122]. The DEIM method approximates a nonlinear function by combining projection with interpolation, constructing interpolation indices that specify an interpolation-based projection to approximate nonlinear terms with a lower computational cost. These methods for generating the POD basis differ in the way nonlinear terms are treated, with the efficacy of each approach depending on the particular problem. We refer to [123] and the references therein for more detail on POD approaches to 4D-Var.

2.2.3 | Balanced truncation within 4D-Var

An alternative method for constructing a reduced order model as a simplification for use within the inner loop of incremental 4D-Var is proposed in [23, 24, 84, 85]. The authors apply balanced truncation [96] to project the model and observation operators onto a lower dimensional space, and hence the resulting minimisation takes

place in a space of reduced dimension.

These papers consider strong constraint 4D-Var to set up the linear system used for balanced truncation. In Chapter 5 we extend these ideas to weak constraint 4D-Var, and consider the efficacy of this method compared to other projection methods. A limitation of this approach is that it is designed for linear models, with the balanced truncation method requiring a stable, linear system.

In this chapter we have presented a short review of the existing methodology for applying some form of model order reduction to the data assimilation problem. There have been many different approaches for achieving a reduction in the complexity of both the Kalman filter and variational methods for data assimilation, with some hybrid methods which combine ideas and results from both. These methods have included constructing ensembles to generate Monte Carlo estimations for the covariance matrices and using ideas from control theory, such as the balanced truncation method for model reduction. As listed in Chapter 1, there are several approaches for applying data assimilation, and as a result this literature review may be incomplete, however it allows us to place the new methods introduced in the subsequent chapters of this thesis in the wider context of the existing literature.

CHAPTER 3

A LOW-RANK APPROACH TO WEAK CONSTRAINT 4D-VAR

The work in this chapter is the basis of the paper [55] which appeared in Journal of Computational Physics 357 (2018), pp. 263-281.

3.1 | Introduction

As mentioned in Chapter 1, data assimilation is used in many applications including numerical weather prediction and other geosciences to combine a numerical model with observations obtained from a physical system, in order to create a more accurate estimate for the true state of the system.

A property which these applications all share is the vast dimensionality of the state vectors involved. In numerical weather prediction the systems have variables of order 10^8 [82]. In addition to the requirement that these computations to be solved quickly, the storage requirement presents an obstacle. In this chapter we propose an approach for implementing the weak four-dimensional variational data assimilation method with a low-rank solution in order to achieve a reduction in storage space as well as computation time. The approach investigated here is based on a recent paper [125] which implemented this method in the setting of PDE-constrained optimisation. We introduce here a low-rank modification to GMRES in order to generate low-rank solutions in the setting of data assimilation.

This method was motivated by recent developments in the area of solving large sparse matrix equations, see [12, 77, 99, 110, 117, 118], notably the Lyapunov equation

$$AX + XA^T = -BB^T$$

in which we solve for the square matrix X, where A, B and X are large matrices of conforming dimensions. It is known that if the right hand side of these matrix equations are low-rank, there exist low-rank approximations to X [62]. There are a number of methods which iteratively generate low-rank solutions; see e.g. [44, 86, 99, 110, 117], and it is these ideas which are employed in this chapter.

Alternative methods as discussed in Chapter 2 have been considered for computing low-rank solutions within the data assimilation problem, or considering reduced-order models. In this chapter we take a different approach, the data assimilation problem is considered in its full formulation, however the expensive solve of the linear system is done in a low-rank in time framework.

In the next section we introduce a saddle point formulation of weak constraint four dimensional variational data assimilation. Section 3.2 explains the connection between the arising linear system and the solution to matrix equations. We then introduce a low-rank approach to GMRES. Numerical results are presented in Section 3.3, with an extension to time-dependent systems considered in Section 3.4.

3.2 | Low-rank approach

The approach we take here considers the incremental implementation of weak constraint 4D-Var. As mentioned in Section 1.3.2, the incremental approach [35] is merely a form of Gauss-Newton iteration and generates an approximation to the solution of $x = \text{argmin } J(x)$, where J is the weak 4D-Var cost function (1.10). It has been shown that if a full resolution linearisation is used this is not necessarily an approximation, and returns an exact solution [64, 83].

We approximate the 4D-Var cost function by a quadratic function of an increment $\delta x^{(\ell)} = \left[(\delta x_0^{(\ell)})^T, (\delta x_1^{(\ell)})^T, \ldots, (\delta x_N^{(\ell)})^T\right]^T$ defined as

$$\delta x^{(\ell)} = x^{(\ell+1)} - x^{(\ell)}, \tag{3.1}$$

where $x^{(\ell)} = \left[(x_0^{(\ell)})^T, (x_1^{(\ell)})^T, \ldots, (x_N^{(\ell)})^T\right]^T$ denotes the ℓ-th iterate of the Gauss-Newton algorithm. Updating this estimate is implemented in an *outer loop*, whilst generating $\delta x^{(\ell)}$ is referred to as the *inner loop*. This increment $\delta x^{(\ell)}$ is a solution

to the minimisation of the linearised cost function

$$\begin{aligned}\tilde{J}(\delta x^{(\ell)}) = &\frac{1}{2}(\delta x_0^{(\ell)} - b_0^{(\ell)})^T B^{-1}(\delta x_0^{(\ell)} - b_0^{(\ell)}) \\ &+ \frac{1}{2}\sum_{k=0}^{N}(d_k^{(\ell)} - H_k \delta x_k^{(\ell)})^T R_k^{-1}(d_k^{(\ell)} - H_k \delta x_k^{(\ell)}) \\ &+ \frac{1}{2}\sum_{k=1}^{N}(\delta x_k^{(\ell)} - M_k \delta x_{k-1}^{(\ell)} - c_k^{(\ell)})^T Q_k^{-1}(\delta x_k^{(\ell)} - M_k \delta x_{k-1}^{(\ell)} - c_k^{(\ell)}).\end{aligned} \tag{3.2}$$

Here $M_k \in \mathbb{R}^{n\times n}$ and $H_k \in \mathbb{R}^{p_k \times n}$, are linearisations of $\mathcal{M}_k$ and $\mathcal{H}_k$ about the current state trajectory $x^{(\ell)}$. For convenience and conciseness, we introduce

$$b_0^{(\ell)} = x_0^b - x_0^{(\ell)}, \tag{3.3}$$

$$d_k^{(\ell)} = y_k - \mathcal{H}_k(x_k^{(\ell)}), \tag{3.4}$$

$$c_k^{(\ell)} = \mathcal{M}_k(x_{k-1}^{(\ell)}) - x_k^{(\ell)}. \tag{3.5}$$

We define the following vectors in order to rewrite the cost function in a more compact form:

$$\delta x = \begin{bmatrix}\delta x_0 \\ \delta x_1 \\ \vdots \\ \delta x_N\end{bmatrix}, \quad \delta p = \begin{bmatrix}\delta x_0 \\ \delta q_1 \\ \vdots \\ \delta q_N\end{bmatrix},$$

where we have dropped the superscript for the outer loop iteration. These two vectors are related by $\delta q_k = \delta x_k - M_k \delta x_{k-1}$, or in matrix form

$$\delta p = \mathbf{L}\delta x, \tag{3.6}$$

where

$$\mathbf{L} = \begin{bmatrix} I & & & \\ -M_1 & I & & \\ & \ddots & \ddots & \\ & & -M_N & I \end{bmatrix} \in \mathbb{R}^{(N+1)n\times(N+1)n}. \tag{3.7}$$

Furthermore, we introduce the following matrices:

$$\mathbf{D} = \begin{bmatrix} B & & & \\ & Q_1 & & \\ & & \ddots & \\ & & & Q_N \end{bmatrix} \in \mathbb{R}^{(N+1)n\times(N+1)n}, \tag{3.8}$$

$$\mathbf{R} = \begin{bmatrix} R_0 & & & \\ & R_1 & & \\ & & \ddots & \\ & & & R_N \end{bmatrix} \in \mathbb{R}^{\sum_{k=0}^{N} p_k \times \sum_{k=0}^{N} p_k}, \tag{3.9}$$

$$\mathbf{H} = \begin{bmatrix} H_0 & & & \\ & H_1 & & \\ & & \ddots & \\ & & & H_N \end{bmatrix} \in \mathbb{R}^{\sum_{k=0}^{N} p_k \times (N+1)n}, \tag{3.10}$$

and vectors

$$b = \begin{bmatrix} b_0 \\ c_1 \\ \vdots \\ c_N \end{bmatrix} \in \mathbb{R}^{(N+1)n}, \quad \text{and} \quad d = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_N \end{bmatrix} \in \mathbb{R}^{\sum_{k=0}^{N} p_k}. \tag{3.11}$$

This representation allows us to write (3.2), with the superscripts dropped, as a function of δx:

$$\tilde{J}(\delta x) = \frac{1}{2}(\mathbf{L}\delta x - b)^T \mathbf{D}^{-1}(\mathbf{L}\delta x - b) + \frac{1}{2}(\mathbf{H}\delta x - d)^T \mathbf{R}^{-1}(\mathbf{H}\delta x - d). \tag{3.12}$$

Minimising the cost function is equivalent to setting the gradient of the cost function to be zero, and solving the resulting linear system. Indeed, taking the gradient of this cost function with respect to δx, the resulting equation is

$$\nabla \tilde{J}(\delta x) = \mathbf{L}^T \mathbf{D}^{-1}(\mathbf{L}\delta x - b) + \mathbf{H}^T \mathbf{R}^{-1}(\mathbf{H}\delta x - d) = 0. \tag{3.13}$$

Defining $\lambda = \mathbf{D}^{-1}(b - \mathbf{L}\delta x)$ and $\mu = \mathbf{R}^{-1}(d - \mathbf{H}\delta x)$, allows us to write the gradient at the minimum as

$$\nabla \tilde{J} = \mathbf{L}^T \lambda + \mathbf{H}^T \mu = 0. \tag{3.14}$$

Additionally, we have

$$\mathbf{D}\lambda + \mathbf{L}\delta x = b, \tag{3.15}$$

$$\mathbf{R}\mu + \mathbf{H}\delta x = d, \tag{3.16}$$

and (3.14), (3.15) and (3.16) can be combined into a single linear system:

$$\begin{bmatrix} \mathbf{D} & 0 & \mathbf{L} \\ 0 & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^T & \mathbf{H}^T & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \mu \\ \delta x \end{bmatrix} = \begin{bmatrix} b \\ d \\ 0 \end{bmatrix}, \tag{3.17}$$

which is to be solved to obtain δx.

This equation is known as the saddle-point formulation for weak constraint 4D-Var, and allows us to exploit the saddle point structure for linear solves and corresponding preconditioning techniques [14, 17, 125].

The saddle point matrix in (3.17), is a square symmetric indefinite matrix of size $\left(2n(N+1) + \sum_{k=0}^{N} p_k\right)$. In order to successfully solve this system we must use an iterative solver such as MINRES (minimal residual method) [98] or GMRES (generalised minimal residual) [111] as it is infeasible with these large problem sizes to use a direct method.

MINRES and GMRES are both Krylov subspace methods for solving linear systems $Ax = b$. These methods obtain an approximate solution x_k from a Krylov subspace

$$\mathcal{K}_k(A, b) = \text{span}\{b, Ab, A^2b, \ldots, A^{k-1}b\},$$

by imposing the Petrov-Galerkin condition $b - Ax_k \perp \mathcal{L}_k$, where $\mathcal{L}_k$ is another subspace of size k. The approximate solutions generated through MINRES and GMRES are such that the norm of the residual $\|b - Ax_k\|$ is minimised.

The MINRES and GMRES methods are derived from the Lanczos and Arnoldi algorithms respectively. Hence MINRES can only be used for symmetric systems whilst GMRES can be used for non-symmetric cases.

When solving problems with an iterative solver, we additionally require a good choice of preconditioner. This is typically to improve the condition number of the matrix A, and hence convergence of the iterative method [88]. There are many preconditioners designed for saddle point systems [14, 15, 16, 17, 18, 53], however in a data assimilation setting, the saddle point matrix has different properties to majority of other saddle point problems in the literature. We refer to Chapter 4 for greater discussion on this topic. The inexact constraint preconditioner [17] has been found to be an effective choice of preconditioner for the data assimilation problem [53], but application of this results in a non-symmetric system necessitating the use of GMRES.

Furthermore, to overcome the storage requirements of the matrix in (3.17), we wish to avoid forming it (and indeed as many of the submatrices as possible), which

motivates the method described in the remainder of this chapter.

3.2.1 | Kronecker formulation

As noted above, the matrix formed in the saddle point formulation is very large, as indeed are the vectors $\lambda, \mu, \delta x$. We wish to adapt the ideas developed in [125] in order to solve (3.17). This approach is dependent on the Kronecker product and the vec $(\cdot)$ operator; which are defined to be

$$\mathcal{A} \otimes \mathcal{B} = \begin{bmatrix} a_{11}\mathcal{B} & \cdots & a_{1n}\mathcal{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathcal{B} & \cdots & a_{mn}\mathcal{B} \end{bmatrix}, \quad \text{vec}\,(\mathcal{C}) = \begin{bmatrix} c_{11} \\ \vdots \\ c_{1n} \\ \vdots \\ c_{mn} \end{bmatrix}.$$

We also make use of the relationship between the two:

$$(\mathcal{B}^T \otimes \mathcal{A})\text{vec}\,(\mathcal{C}) = \text{vec}\,(\mathcal{A}\mathcal{C}\mathcal{B})\,. \tag{3.18}$$

Employing these definitions, we may rewrite (3.17) as

$$\begin{bmatrix} E_1 \otimes B + E_2 \otimes Q & 0 & I_{N+1} \otimes I_n + C \otimes M \\ 0 & I_{N+1} \otimes R & I_{N+1} \otimes H \\ I_{N+1} \otimes I_n + C^T \otimes M^T & I_{N+1} \otimes H^T & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \mu \\ \delta x \end{bmatrix} = \begin{bmatrix} b \\ d \\ 0 \end{bmatrix}, \tag{3.19}$$

where we make the additional assumptions that $Q_k = Q$, $R_k = R$, $H_k = H$, $M_k = M$ and the number of observations $p_k = p$ for each k. The extended case relaxing this assumption is considered in Section 3.4. Here

$$C = \begin{bmatrix} 0 & & & \\ -1 & 0 & & \\ & \ddots & \ddots & \\ & & -1 & 0 \end{bmatrix}, \quad E_1 = \begin{bmatrix} 1 & & & \\ & 0 & & \\ & & \ddots & \\ & & & 0 \end{bmatrix}, \text{ and } E_2 = \begin{bmatrix} 0 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}.$$

The matrices $C, E_1, E_2, I_{N+1} \in \mathbb{R}^{(N+1)\times(N+1)}$, whilst $B, Q, M, I_n \in \mathbb{R}^{n\times n}$, $H \in \mathbb{R}^{p\times n}$, and $R \in \mathbb{R}^{p\times p}$, where n is the size of the state space, N the number of timesteps in the assimilation window, and p is the number of observations.

Using (3.18), we may rewrite (3.19) as the simultaneous matrix equations:

$$\begin{aligned} B\Lambda E_1 + Q\Lambda E_2 + X + MXC^T &= \mathbb{b}, \\ RU + HX &= \mathbb{d}, \\ \Lambda + M^T\Lambda C + H^T U &= 0., \end{aligned} \tag{3.20}$$

where we suppose $\lambda, \delta x, b, \mu$ and d are vectorised forms of the matrices $\Lambda, X, \mathbb{b} \in \mathbb{R}^{n\times(N+1)}$ and $U, \mathbb{d} \in \mathbb{R}^{p\times(N+1)}$ respectively. The three equations (3.20) are generalised Sylvester equations, which we solve for Λ, U and X, though to update the state estimate in incremental data assimilation, we require only δx and hence the solution X.

For standard Sylvester equations of the form $\mathcal{A}\mathcal{X} + \mathcal{X}\mathcal{B} = \mathcal{C}$, it is known that if the right hand side $\mathcal{C}$ is low-rank, then there exist low-rank approximate solutions [62]. Indeed, recent algorithms for solving these Sylvester equations have focused on constructing low-rank approximate solutions. These algorithms include Krylov subspace methods (see [118]) and ADI (alternating direction implicit) based methods (see [10, 13, 54]). It is this knowledge which motivates the following approach.

3.2.2 | Existence of a low-rank solution

In this section, we wish to show that there exist low-rank approximate solutions to the weak constraint variational data assimilation problem as in the setting above. To do so, we consider the tensor rank of δx.

Definition (Tensor rank). *Let $x = \mathrm{vec}\,(X) \in \mathbb{R}^{n^2}$. The minimal number r such that*

$$x = \sum_{i=1}^{r} u_i \otimes v_i, \tag{3.21}$$

where $u_i, v_i \in \mathbb{R}^n$ is called the tensor rank *of the vector x.*

We now state some properties of the tensor rank.

Lemma 3.1. *Let $x \in \mathbb{R}^{n^2}$ be the vectorisation of $X \in \mathbb{R}^{n\times n}$, such that $x = \mathrm{vec}\,(X)$. The tensor rank of the vector x is equal to the rank of the matrix X.*

Proof. Let X have rank r, thus X can be decomposed as

$$X = \sum_{i=1}^{r} v_i u_i^T.$$

Vectorising this matrix we obtain

$$x = \mathrm{vec}\,(X) = \sum_{i=1}^{r} \mathrm{vec}\left(v_i u_i^T\right),$$

and applying the identity (3.18): $(\mathcal{B}^T \otimes \mathcal{A})\mathrm{vec}\,(\mathcal{C}) = \mathrm{vec}\,(\mathcal{A}\mathcal{C}\mathcal{B})$ with $\mathcal{C}$ the scalar 1 we obtain:

$$x = \mathrm{vec}\,(X) = \sum_{i=1}^{r} u_i \otimes v_i,$$

as desired. □

Lemma 3.2. *Let $x \in \mathbb{R}^{nm}$ be the vectorisation of $X \in \mathbb{R}^{n\times m}$ with tensor rank r, and let $\mathbf{A} \in \mathbb{R}^{nm\times nm}$ be of the form*

$$\mathbf{A} = \sum_{i=1}^{k} (A_i \otimes B_i), \tag{3.22}$$

where $A_i \in \mathbb{R}^{n\times n}$, $B_i \in \mathbb{R}^{m\times m}$ for $i = 1 \dots k$. The tensor rank of the vector $\mathbf{A}x$ is at most kr.

Furthermore if $\mathbf{B} = \sum_{j=1}^{\ell} (C_j \otimes D_j) \in \mathbb{R}^{nm\times nm}$ with $C_j \in \mathbb{R}^{n\times n}$, $D_j \in \mathbb{R}^{m\times m}$ for $j = 1 \dots \ell$. The tensor rank of the vector $\mathbf{B}\mathbf{A}x$ is at most ℓkr.

Proof. Using the identity (3.18), we may rewrite $\mathbf{A}x$ as

$$\sum_{i=1}^{k} (A_i \otimes B_i)x = \mathrm{vec}\left(\sum_{i=1}^{k} B_i X A_i^T\right).$$

Using familiar properties of the rank of a matrix we observe

$$\begin{aligned}
\mathrm{rank}\left(\sum_{i=1}^{k} B_i X A_i^T\right) &\leq \sum_{i=1}^{k} \mathrm{rank}(B_i X A_i^T) \\
&\leq \sum_{i=1}^{k} \min\{\mathrm{rank}(B_i), \mathrm{rank}(X), \mathrm{rank}(A_i)\} \\
&\leq kr.
\end{aligned}$$

Applying Lemma 3.1, the rank of $\sum_{i=1}^{k} B_i X A_i^T$ is equivalent to the tensor rank of $\sum_{i=1}^{k} (A_i \otimes B_i)x$, and hence this vector has tensor rank at most kr.

Considering the product $\mathbf{B}\mathbf{A}$, we obtain a matrix which is the sum of ℓk Kronecker products. Applying the previous yields the desired result. □

Remark. *Because of this result, we sometimes refer to a matrix of the form* (3.22)

as a matrix of tensor rank k.

In order to consider the existence of a low-rank approximate solution, we make use of the following results from [61] and the method used in [9] for considering low-rank solutions to problems with a tensor structure. In [61] it is shown that for a stable matrix $\mathcal{A}$, with eigenvalues in the left complex half plane, the inverse of $\mathcal{A}$ is given by $-\int_0^\infty \exp(t\mathcal{A})\mathrm{d}t$, since

$$\mathcal{A}\left(-\int_0^\infty \exp(t\mathcal{A})\mathrm{d}t\right) = -\int_0^\infty \frac{\partial}{\partial t}\exp(t\mathcal{A})\mathrm{d}t = \exp(0\mathcal{A}) = I,$$

due to the negative eigenvalues of $\mathcal{A}$.

The integral in the inverse can be approximated by quadrature, and we can apply the following Lemma from [61].

Lemma 3.3. *[61] Let $\mathcal{A}$ be a matrix with the spectrum $\sigma(\mathcal{A})$ contained in a rectangle Ω in $\mathbb{C}_-$, and let Γ denote the boundary of a rectangle which encloses this such that the distance from Γ to $\sigma(\mathcal{A})$ is at least 1. For each $k \in \mathbb{N}$ define the following quadrature points and weights [124]:*

$$\begin{aligned} h_{st} &:= \frac{\pi^2}{\sqrt{k}}, \\ t_j &:= \log(\exp(jh_{st}) + \sqrt{1+\exp(2jh_{st})}), \\ w_j &:= \frac{h_{st}}{\sqrt{1+\exp(-2jh_{st})}}. \end{aligned}$$

Then there exists a constant C_{st} independent of $\mathcal{A}$ and k such that for an arbitrary matrix norm,

$$\left\| \int_0^\infty \exp(t\mathcal{A}) - \sum_{j=-k}^{k} w_j \exp(t_j\mathcal{A}) \right\| \leq \frac{C_{st}}{2\pi}\exp\left(\frac{\mu+1}{\pi} - \pi\sqrt{2k}\right) \oint_\Gamma \|(\gamma I - \mathcal{A})^{-1}\|\mathrm{d}_\Gamma\gamma, \tag{3.23}$$

where $\mu \geq |\mathrm{Im}(\lambda)|$ for all $\lambda \in \sigma(\mathcal{A})$.

It has been noted that the constant C_{st} is problem independent, and has been experimentally determined as $C_{st} \approx 2.75$, see [77].

We observe that taking a larger choice of k, and thus more quadrature points, the smaller the error due to the $\exp(\frac{\mu+1}{\pi} - \pi\sqrt{2k})$ term.

For matrices of the form $\mathcal{A} = A_1 \otimes I + I \otimes A_2$, or more generally,

$$\mathcal{A} = \sum_{i=1}^{d} \hat{A}_i, \qquad \hat{A}_i = \underbrace{I \otimes \cdots \otimes I}_{i-1 \text{ terms}} \otimes A_i \otimes \underbrace{I \otimes \cdots \otimes I}_{d-i \text{ terms}}, \quad A_i \in \mathbb{R}^{n\times n}, \tag{3.24}$$

tensor matrices with eigenvalues in the left complex half plane, we can consider the matrix exponential and obtain the following lemma.

Lemma 3.4. *Let $\mathcal{A}$ be a matrix of tensor structure* (3.24)*, then*

$$\exp(\mathcal{A}) = \exp(A_1) \otimes \cdots \otimes \exp(A_i) \otimes \cdots \otimes \exp(A_d). \tag{3.25}$$

Proof. For illustration we take the case $d = 2$ with any matrices $A_1, A_2 \in \mathbb{R}^{n\times n}$,

$$(A_1 \otimes I)^n = A_1^n \otimes I, \qquad (I \otimes A_2)^n = I \otimes A_2^n.$$

Thus, considering the Taylor series expansion of the matrix exponential,

$$\exp(A_1 \otimes I) = \exp(A_1) \otimes I, \qquad \exp(I \otimes A_2) = I \otimes \exp(A_2),$$

and hence

$$\begin{aligned}\exp(\mathcal{A}) = \exp(A_1 \otimes I + I \otimes A_2) &= \exp(A_1 \otimes I)\exp(I \otimes A_2)\\ &= (\exp(A_1) \otimes I)(I \otimes \exp(A_2))\\ &= \exp(A_1) \otimes \exp(A_2).\end{aligned}$$

The extension to $d > 2$ follows similarly. □

Combining the results of Lemma 3.4 and Lemma 3.3 we can state the existence of an approximate inverse to $\mathcal{A}$, with an error bound between the approximate and exact inverse.

Lemma 3.5. *[61] Let $\mathcal{A}$ be a matrix of tensor structure* (3.24) *with $d = 2$ and the spectrum $\sigma(\mathcal{A})$ contained in a rectangle Ω in $\mathbb{C}_-$, and let Γ denote the boundary of a rectangle which encloses this such that the distance from Γ to $\sigma(\mathcal{A})$ is at least 1. Let $k \in \mathbb{N}$, and t_j, w_j denote the points and weights from* Lemma *3.3. Then the inverse $\mathcal{A}^{-1}$ of $\mathcal{A}$ can be approximated by*

$$\widetilde{\mathcal{A}^{-1}} \coloneqq -\sum_{j=-k}^{k} w_j \exp(t_j A_1) \otimes \exp(t_j A_2), \tag{3.26}$$

with the approximation error

$$\left\|\mathcal{A}^{-1} - \widetilde{\mathcal{A}^{-1}}\right\| \leq \frac{C_{st}}{2\pi} \exp\left(\frac{\mu+1}{\pi} - \pi\sqrt{2k}\right) \oint_\Gamma \|(\gamma I - \mathcal{A})^{-1}\| \mathrm{d}_\Gamma \gamma, \tag{3.27}$$

where $\mu \geq |\mathrm{Im}(\lambda)|$ for all $\lambda \in \sigma(\mathcal{A})$.

If we consider the matrix $\mathbf{L} = I \otimes I + C \otimes M$ from (3.7), let us rewrite this as $\mathbf{L} = (I \otimes -M)(-C \otimes I + I \otimes -M^{-1})$. The matrix $(-C \otimes I + I \otimes -M^{-1})$ satisfies the structure of (3.24) for $d = 2$, and thus we can apply Lemma 3.5 to obtain an approximation to $\mathbf{L}^{-1}$ with the above error bound dependent on the number of quadrature points.

We are now ready to state our result on the existence of low-rank solutions to the weak constraint 4D-Var cost function (3.12).

Theorem 3.6. *Consider the problem* (3.12)*, let the model and observations be time-independent, with* $M = M_k, R = R_k, H = H_k, Q = Q_k$ *for all* k*. Furthermore, assume* M *is invertible, and the spectrum of* $(-C \otimes I + I \otimes -M^{-1})$ *is contained in a rectangle in* $\mathbb{C}_-$*. Then the minimum of the cost function* (3.12)*,* δx *can be approximated by a vector of tensor rank at most* $4(2r+1)^2(\mathrm{rank}(b)+p+1)$*. Here* r *arises from the quadrature approximation in* Lemma *3.5,* b *is the background term from* (3.11) *and* p *is the number of observations in the data assimilation problem.*

This approximation $\widetilde{\delta x}$ *is of the form*

$$\widetilde{\delta x} \coloneqq \widetilde{\mathbf{L}^{-1}}\mathbf{D}(\widetilde{\mathbf{L}^{-1}})^T(-f + \mathbf{H}^T g), \tag{3.28}$$

where

$$\widetilde{\mathbf{L}^{-1}} = \sum_{j=-r}^{r} w_j \exp(-C) \otimes \exp\left(M^{-1}\right) M^{-1}, \tag{3.29}$$

with t_j *and* w_j *the quadrature points and weights as defined in* Lemma *3.3. The vectors* f *and* g *are the right hand side of the normal equations* (3.13)*:*

$$f \coloneqq \mathbf{L}^T\mathbf{D}^{-1}b + \mathbf{H}^T\mathbf{R}^{-1}d, \tag{3.30}$$

and the solution of

$$(\mathbf{I} + \mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}\mathbf{H}^T)g = \mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}f \tag{3.31}$$

respectively.

Proof. Let us consider the normal equations (3.13) which can be written

$$\underbrace{(\mathbf{L}^T\mathbf{D}^{-1}\mathbf{L} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})}_{\coloneqq \mathbf{S}}\,\delta x = \underbrace{\mathbf{L}^T\mathbf{D}^{-1}b + \mathbf{H}^T\mathbf{R}^{-1}d}_{\coloneqq f}. \tag{3.32}$$

We denote the matrix $\mathbf{S}$, highlighting that this is (minus) the Schur Complement of the saddle point system (3.17), and the right hand side of (3.32) as f.

Applying the Sherman-Morrison-Woodbury formula [60, 70] we obtain

$$\mathbf{S}^{-1} = -\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T} + \mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}\mathbf{H}^T(\mathbf{I} + \mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}\mathbf{H}^T)^{-1}\mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}$$

and thus

$$\begin{aligned}\delta x &= \mathbf{S}^{-1}f \\ &= \mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}(-f + \mathbf{H}^T(\mathbf{I} + \mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}\mathbf{H}^T)^{-1}\mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}f). \quad (3.33)\end{aligned}$$

Let $g = (\mathbf{I} + \mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}\mathbf{H}^T)^{-1}\mathbf{R}^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}f$, and $G = \mathrm{vec}^{-1}(g)$, then recalling that $\mathbf{H} = I \otimes H$ we can write

$$\begin{aligned}\mathrm{vec}^{-1}\left(\mathbf{H}^T g\right) &= \mathrm{vec}^{-1}\left((I \otimes H^T)g\right) \\ &= H^T G \\ &= \sum_{j=1}^{p} (H_j)^T G_j.\end{aligned}$$

Here we use the subscript j to denote the j-th row, noting that H is an $p \times n$ matrix. It follows that

$$\mathbf{H}^T g = \sum_{j=1}^{p} (G_j)^T \otimes (H_j)^T, \quad (3.34)$$

and hence returning to (3.33) we obtain

$$\begin{aligned}\delta x &= \mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}(-f + \mathbf{H}^T g) \quad (3.35)\\ &= \mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}(-f + \sum_{j=1}^{p} (G_j)^T \otimes (H_j)^T). \quad (3.36)\end{aligned}$$

To consider the tensor rank of δx, we consider the individual components. It follows from the definition of tensor rank that $\mathbf{H}^T g$ is rank p, as we have a sum of p Kronecker products. We can decompose f as

$$\begin{aligned}f &= \mathbf{L}^T\mathbf{D}^{-1}b + \mathbf{H}^T\mathbf{R}^{-1}d \\ &= (I \otimes I + C \otimes M)^T(E_1 \otimes B^{-1} + E_2 \otimes Q^{-1})b + (I \otimes H^T R^{-1})d,\end{aligned}$$

where b and d are the vectors defined in (3.11).

Applying Lemma 3.2, the tensor rank of the first part is bounded by 4 rank(b), as $(I \otimes I + C \otimes M)(E_1 \otimes B^{-1} + E_2 \otimes Q^{-1})$ contains four terms. However $\mathrm{E}_1 = 1$ and hence the tensor rank of this term is more tightly bounded by $(2 + 2\ \mathrm{rank}(b))$. Again

applying Lemma 3.2, the second term $(I \otimes H^T R^{-1})d$ has at most the same tensor rank as the vector d. Furthermore, since d is obtained from our observations, it has at most rank p (the number of observations at each timestep). Thus the tensor rank of $(-f + \mathbf{H}^T g)$ is at most $(2\,\text{rank}(b) + 2p + 2)$.

It remains to investigate the tensor rank of $\widetilde{\mathbf{L}^{-1}}\mathbf{D}(\widetilde{\mathbf{L}^{-1}})^T \approx \mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}$. Let us rewrite $\mathbf{L} = I \otimes I + C \otimes M = (I \otimes -M)(-C \otimes I + I \otimes -M^{-1})$. The matrix $(-C \otimes I + I \otimes -M^{-1})$ satisfies the structure of (3.24), and thus we can apply Lemma 3.5. The inverse of $\mathbf{L} = (I \otimes -M)(-C \otimes I + I \otimes -M^{-1})$ can hence be approximated by

$$\widetilde{\mathbf{L}^{-1}} = \sum_{j=-r}^{r} w_j \exp(-C) \otimes \exp\left(M^{-1}\right) M^{-1}.$$

From this we see that the approximation $\widetilde{\mathbf{L}^{-1}}$ has a tensor rank of $(2r+1)$ which arises from the quadrature. Thus, since $\mathbf{D} = (E_1 \otimes B + E_2 \otimes Q)$ is of tensor rank 2, the approximation $\widetilde{\mathbf{L}^{-1}}\mathbf{D}(\widetilde{\mathbf{L}^{-1}})^T$ is of tensor rank $2(2r+1)^2$.

Therefore, applying Lemma 3.2, we consider the tensor rank of $(-f + \mathbf{H}^T g)$, $2(\text{rank}(b) + p + 1)$ and that of $\widetilde{\mathbf{L}^{-1}}\mathbf{D}(\widetilde{\mathbf{L}^{-1}})^T$ which is $2(2r+1)^2$ we obtain the result that an approximation $\widetilde{\delta x}$ of the form

$$\widetilde{\delta x} = \widetilde{\mathbf{L}^{-1}}\mathbf{D}(\widetilde{\mathbf{L}^{-1}})^T(-f + \mathbf{H}^T g), \tag{3.37}$$

has a tensor rank of at most $4(2r+1)^2(\text{rank}(b) + p + 1)$. □

We have therefore shown that low-rank approximate solutions to the weak constraint variational data assimilation problem do exist. The method we illustrated in this proof of existence, using quadrature is not the approach we take for generating low-rank solutions in the remainder of the chapter, however does provide some insight into the properties of low-rank solutions. Here the rank of the solution is related to the number of observations taken, and the tensor rank of our background vector b. If these are both small, there is a greater chance of observing a low-rank approximation solution. In applications, the number of observations taken each timestep is significantly lower than the size of the state space vector, however less can be said about the tensor rank of the background vector.

We had to make a number of assumptions to obtain this result, including that the model and observations are time-independent. However, as we see experimentally in Section 3.4, relaxing this assumption still results in low-rank approximate solutions.

Let us now consider a method for obtaining low-rank approximate solutions in practice.

3.2.3 | Low-rank GMRES (LR-GMRES)

In order to find low-rank approximate solutions, we suppose as in [9, 125], that the matrices Λ, U, X in (3.20) have low-rank representations, with

$$\Lambda = W_\Lambda V_\Lambda^T, \qquad W_\Lambda \in \mathbb{R}^{n\times k_\Lambda}, V_\Lambda \in \mathbb{R}^{(N+1)\times k_\Lambda}, \tag{3.38}$$

$$U = W_U V_U^T, \qquad W_U \in \mathbb{R}^{p\times k_U}, V_U \in \mathbb{R}^{(N+1)\times k_U}, \tag{3.39}$$

$$X = W_X V_X^T, \qquad W_X \in \mathbb{R}^{n\times k_X}, V_X \in \mathbb{R}^{(N+1)\times k_X}, \tag{3.40}$$

where $k_\Lambda, k_U, k_X \ll n, N$. This allows us to rewrite (3.20) as follows:

$$\begin{aligned}
\begin{bmatrix} BW_\Lambda & QW_\Lambda & W_X & MW_X \end{bmatrix} \begin{bmatrix} V_\Lambda^T E_1 \\ V_\Lambda^T E_2 \\ V_X^T \\ V_X^T C^T \end{bmatrix} &= \mathbb{b}, \\
\begin{bmatrix} RW_U & HW_X \end{bmatrix} \begin{bmatrix} V_U^T \\ W_X^T \end{bmatrix} &= \mathbb{d}, \\
\begin{bmatrix} W_\Lambda & M^T W_\Lambda & H^T W_U \end{bmatrix} \begin{bmatrix} V_\Lambda^T \\ V_\Lambda^T C \\ V_U^T \end{bmatrix} &= 0.
\end{aligned} \tag{3.41}$$

Since using a direct solver would be infeasible, we use an iterative solver, in this case GMRES [111] to allow for flexibility in choosing a preconditioner, see Chapter 4. Algorithm 1 details a low-rank implementation of GMRES, which leads to low-rank approximate solutions to (3.19), using (3.41). Fundamentally this is the same as a traditional vector-based GMRES with a vector w, where instead here we have

$$\begin{bmatrix} \lambda \\ \mu \\ \delta x \end{bmatrix} = \operatorname{vec}\left(\begin{bmatrix} W_\Lambda V_\Lambda^T \\ W_U V_U^T \\ W_X V_X^T \end{bmatrix} \right) = \operatorname{vec}\left(\begin{bmatrix} W_{11} W_{12}^T \\ W_{21} W_{22}^T \\ W_{31} W_{32}^T \end{bmatrix} \right) = w,$$

introducing the notation W_{k1} and W_{k2} for $k = 1, 2, 3$ to ensure consistent notation in the intermediate steps of LR-GMRES.

To apply the vector addition $x = y + \eta z$ for some scalar η within LR-GMRES, we observe that this is equivalent to applying the concatenation $X_{k1} = [Y_{k1}, \quad \eta Z_{k1}]$,

$X_{k2} = [Y_{k2}, \quad Z_{k2}]$ for $k = 1, 2, 3$, since $X_{k1}X_{k2}^T = Y_{k1}Y_{k2}^T + \eta Z_{k1}Z_{k2}^T$ and hence

$$x = \operatorname{vec}\left(\begin{bmatrix} X_{11}X_{12}^T \\ X_{21}X_{22}^T \\ X_{31}X_{32}^T \end{bmatrix}\right) = \operatorname{vec}\left(\begin{bmatrix} Y_{11}Y_{12}^T + \eta Z_{11}Z_{12}^T \\ Y_{21}Y_{22}^T + \eta Z_{21}Z_{22}^T \\ Y_{31}Y_{32}^T + \eta Z_{31}Z_{32}^T \end{bmatrix}\right) = y + \eta z.$$

In Algorithm 1, we employ the same notation as in [125], using the brackets $\{\}$ as a concatenation and truncation operation. Furthermore, after applying matrix multiplication or preconditioning, we also truncate the resulting matrices. How this truncation could be implemented is also treated in [125], with options including a truncated singular value decomposition, possibly through Matlab's inbuilt `svds` function, or a skinny QR factorisation. In the numerical results to follow, we use a modification of the Matlab `svds` function.

In order to compute the inner product $\langle w, v^{(i)} \rangle$ which arises in GMRES when computing the entries of the Hessenberg matrix (see line 11 in Algorithm 1), we make use of the relation between the trace and vec operators:

$$\operatorname{trace}(A^T B) = \operatorname{vec}(A)^T \operatorname{vec}(B).$$

Since in this setting, the vectors w and $v^{(i)}$ are the vectorisations

$$\operatorname{vec}\left(\begin{bmatrix} W_{11}W_{12}^T \\ W_{21}W_{22}^T \\ W_{31}W_{32}^T \end{bmatrix}\right) = w \quad \text{and} \quad \operatorname{vec}\left(\begin{bmatrix} V_{11}^{(i)}(V_{12}^{(i)})^T \\ V_{21}^{(i)}(V_{22}^{(i)})^T \\ V_{31}^{(i)}(V_{32}^{(i)})^T \end{bmatrix}\right) = v^{(i)},$$

we see that we may compute the inner product $\langle w, v^{(i)} \rangle$ as

$$\begin{aligned} \langle w, v^{(i)} \rangle \; =& \operatorname{trace}\left((W_{11}W_{12}^T)^T (V_{11}^{(i)}(V_{12}^{(i)})^T)\right) + \operatorname{trace}\left((W_{21}W_{22}^T)^T (V_{21}^{(i)}(V_{22}^{(i)})^T)\right) \\ &+ \operatorname{trace}\left((W_{31}W_{32}^T)^T (V_{31}^{(i)}(V_{32}^{(i)})^T)\right). \end{aligned} \tag{3.42}$$

Importantly however, the matrices formed in (3.42) do not exploit the low-rank nature of the submatrices. Fortunately, using the properties of the trace operator, we may consider instead:

$$\begin{aligned} \langle w, v^{(i)} \rangle \; =& \operatorname{trace}\left(W_{11}^T V_{11}^{(i)}(V_{12}^{(i)})^T W_{12}\right) + \operatorname{trace}\left(W_{21}^T V_{21}^{(i)}(V_{22}^{(i)})^T W_{22}\right) \\ &+ \operatorname{trace}\left(W_{31}^T V_{31}^{(i)}(V_{32}^{(i)})^T W_{32}\right), \end{aligned} \tag{3.43}$$

and hence compute the trace of smaller matrices. This is the method implemented in line 11 of Algorithm 1.

Algorithm 1: Low-rank GMRES (LR-GMRES)

Choose $X_{11}^{(0)}, X_{12}^{(0)}, X_{21}^{(0)}, X_{22}^{(0)}, X_{31}^{(0)}, X_{32}^{(0)}$.
$\{\tilde{X}_{11}, \tilde{X}_{12}, \tilde{X}_{21}, \tilde{X}_{22}, \tilde{X}_{31}, \tilde{X}_{32}\} = \texttt{Amult}(X_{11}^{(0)}, X_{12}^{(0)}, X_{21}^{(0)}, X_{22}^{(0)}, X_{31}^{(0)}, X_{32}^{(0)})$.
$V_{11} = \{B_{11}, \quad -\tilde{X}_{11}\}, \qquad V_{12} = \{B_{12}, \quad \tilde{X}_{12}\},$
$V_{21} = \{B_{21}, \quad -\tilde{X}_{21}\}, \qquad V_{22} = \{B_{22}, \quad \tilde{X}_{22}\},$
$V_{31} = \{B_{31}, \quad -\tilde{X}_{31}\}, \qquad V_{32} = \{B_{32}, \quad \tilde{X}_{32}\}.$
$\xi = [\xi_1, 0, \ldots, 0]$, $\xi_1 = \sqrt{\texttt{traceproduct}(V_{11}^{(1)}, \ldots, V_{11}^{(1)}, \ldots)}$.
for $k = 1, \ldots$ **do**
  $\{Z_{11}^{(k)}, Z_{12}^{(k)}, Z_{21}^{(k)}, Z_{22}^{(k)}, Z_{31}^{(k)}, Z_{32}^{(k)}\} = \texttt{Aprec}(V_{11}^{(k)}, V_{12}^{(k)}, V_{21}^{(k)}, V_{22}^{(k)}, V_{31}^{(k)}, V_{32}^{(k)})$
  $\{W_{11}, W_{12}, W_{21}, W_{22}, W_{31}, W_{32}\} = \texttt{Amult}(Z_{11}^{(k)}, Z_{12}^{(k)}, Z_{21}^{(k)}, Z_{22}^{(k)}, Z_{31}^{(k)}, Z_{32}^{(k)})$.
  for $i = 1, \ldots, k$ **do**
    $h_{i,k} = \texttt{traceproduct}(W_{11}, \ldots, V_{11}^{(i)}, \ldots)$,
    $W_{11} = \{W_{11}, \quad h_{i,k} V_{11}^{(i)}\}, \qquad W_{12} = \{W_{12}, \quad V_{12}^{(i)}\},$
    $W_{21} = \{W_{21}, \quad h_{i,k} V_{21}^{(i)}\}, \qquad W_{22} = \{W_{22}, \quad V_{22}^{(i)}\},$
    $W_{31} = \{W_{31}, \quad h_{i,k} V_{31}^{(i)}\}, \qquad W_{32} = \{W_{22}, \quad V_{32}^{(i)}\}.$
  end for
  $h_{k+1,k} = \sqrt{\texttt{traceproduct}(W_{11}, \ldots, W_{11}, \ldots)}$
  $V_{11}^{(k+1)} = W_{11}/h_{k+1,k}, \qquad V_{12}^{(k+1)} = W_{12},$
  $V_{21}^{(k+1)} = W_{21}/h_{k+1,k}, \qquad V_{22}^{(k+1)} = W_{22},$
  $V_{31}^{(k+1)} = W_{31}/h_{k+1,k}, \qquad V_{32}^{(k+1)} = W_{32}.$
  Apply Givens rotations to kth column of h, i.e.
  for $j = 1, \ldots k-1$ **do**
    $$\begin{bmatrix} h_{j,k} \\ h_{j+1,k} \end{bmatrix} = \begin{bmatrix} c_j & s_j \\ -\bar{s}_j & c_j \end{bmatrix} \begin{bmatrix} h_{j,k} \\ h_{j+1,k} \end{bmatrix}$$
  end for
  Compute kth rotation, and apply to ξ and last column of h.
  $$\begin{bmatrix} \xi_k \\ \xi_{k+1} \end{bmatrix} = \begin{bmatrix} c_k & s_k \\ -\bar{s}_k & c_k \end{bmatrix} \begin{bmatrix} \xi_k \\ 0 \end{bmatrix}, \qquad h_{k,k} = c_k h_{k,k} + s_k h_{k+1,k},$$
  $$h_{k+1,k} = 0.$$
  if $|\xi_{k+1}|$ sufficiently small **then**
    Solve $\tilde{H}\tilde{y} = \xi$, where the entries of $\tilde{H}$ are $h_{i,k}$.
    $Y_{11} = \{\tilde{y}_1 V_{11}^{(1)}, \ \ldots \ , \ \tilde{y}_k V_{11}^{(k)}\}, \qquad Y_{12} = \{\tilde{y}_1 V_{12}^{(1)}, \ \ldots \ , \ \tilde{y}_k V_{12}^{(k)}\}$
    $Y_{21} = \{\tilde{y}_1 V_{11}^{(1)}, \ \ldots \ , \ \tilde{y}_k V_{21}^{(k)}\}, \qquad Y_{22} = \{\tilde{y}_1 V_{22}^{(1)}, \ \ldots \ , \ \tilde{y}_k V_{22}^{(k)}\}$
    $Y_{31} = \{\tilde{y}_1 V_{31}^{(1)}, \ \ldots \ , \ \tilde{y}_k V_{31}^{(k)}\}, \qquad Y_{32} = \{\tilde{y}_1 V_{32}^{(1)}, \ \ldots \ , \ \tilde{y}_k V_{32}^{(k)}\}$
    $\{\tilde{Y}_{11}, \tilde{Y}_{12}, \tilde{Y}_{21}, \tilde{Y}_{22}, \tilde{Y}_{31}, \tilde{Y}_{32}\} = \texttt{Aprec}(Y_{11}, Y_{12}, Y_{21}, Y_{22}, Y_{31}, Y_{32})$
    $X_{11} = \{X_{11}^{(0)}, \ \tilde{Y}_{11}\}, \qquad X_{12} = \{X_{12}^{(0)}, \ \tilde{Y}_{12}\}$
    $X_{21} = \{X_{21}^{(0)}, \ \tilde{Y}_{21}\}, \qquad X_{22} = \{X_{22}^{(0)}, \ \tilde{Y}_{22}\}$
    $X_{31} = \{X_{31}^{(0)}, \ \tilde{Y}_{31}\}, \qquad X_{32} = \{X_{32}^{(0)}, \ \tilde{Y}_{32}\}$
    break
  end if
end for

The matrix vector multiplication Az in traditional GMRES, is implemented in LR-GMRES by considering the low-rank form of the saddle point equations generated in (3.41). The concatenation is explicitly written in Algorithm 2 and is denoted `Amult` in Algorithm 1.

Algorithm 2: Matrix multiplication (`Amult`)

Input: $W_{11}, W_{12}, W_{21}, W_{22}, W_{31}, W_{32}$
Output: $Z_{11}, Z_{12}, Z_{21}, Z_{22}, Z_{31}, Z_{32}$

$$Z_{11} = [BW_{11}, \quad QW_{11}, \quad W_{31}, \quad MW_{31}],$$
$$Z_{12} = [E_1W_{12}, \quad E_2W_{12}, \quad W_{32}, \quad CW_{32}],$$
$$Z_{21} = [RW_{21}, \quad HW_{31}],$$
$$Z_{21} = [W_{22}, \quad W_{32}],$$
$$Z_{31} = [W_{11}, \quad M^TW_{11}, \quad H^TW_{21}],$$
$$Z_{32} = [W_{12}, \quad C^TW_{12}, \quad W_{22}]$$

Note that here we have considered traditional GMRES when implementing LR-GMRES, however it would require only a small modification to allow for restarted GMRES. Preconditioning LR-GMRES is implemented in Algorithm 1 through the `Aprec` function, which works similarly to `Amult` in Algorithm 2 which implements Az from traditional GMRES, here `Aprec` applies $P^{-1}z$ where P approximates A in some sense. This is considered in greater detail in Chapter 4, where we consider the application of preconditioners to this problem.

Due to the truncation steps within the algorithm, introducing a low-rank approximation (by removing small singular values), LR-GMRES does not minimise the residual in the same sense as traditional GMRES. Hence LR-GMRES is more precisely a form of inexact GMRES, see for example [119, 135] and the references therein.

3.3 | Numerical results

In this section we present numerical results using LR-GMRES. (For preconditioning strategies we refer to Chapter 4). We use a maximum iteration number of 20, and stop LR-GMRES when the residual reaches a tolerance of 10^{-6}, or maximum number of iterations is reached. During the algorithm where we truncate the matrices after concatenation and apply `Amult`, we use a truncation tolerance of 10^{-6}. We present examples with different choices of reduced rank r.

3.3.1 | One-dimensional advection-diffusion system

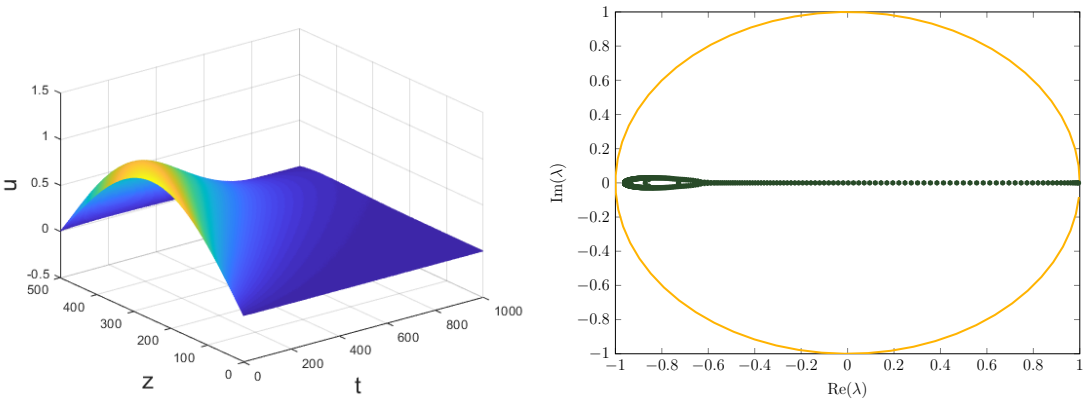
As a first example, let us consider the one-dimensional (linear) advection-diffusion problem, defined as:

$$\frac{\partial}{\partial t}u(z,t) = c_d\frac{\partial^2}{\partial z^2}u(z,t) + c_a\frac{\partial}{\partial z}u(z,t), \tag{3.44}$$

for $z \in [0,1]$, $t \in (0,T)$, subject to the boundary and initial conditions

$$\begin{aligned} u(0,t) = 0, \qquad u(1,t) = 0, \qquad &\in (0,T), \\ u(z,0) = u_0(z), \qquad &z \in [0,1]. \end{aligned}$$

We solve this system with a centered finite difference scheme for u_z and u_t, and a Crank-Nicolson scheme [36] for u_{zz}, discretising z uniformly with $n = 500$, $\Delta z = \frac{1}{499}$ and taking timesteps of size $\Delta t = 10^{-3}$. For this example, we set the underlying system to have $c_d = 0.1$, $c_a = 1.4$ and for the initial condition we take $u_0(z) = \sin(\pi z)$.



A. Advection-diffusion example

B. Eigenvalues λ of M

Figure 3.1: The advection-diffusion example for 1000 timesteps, and the eigenvalues of the model operator M

In Figure 3.1 we see the model evolved forward for 1000 timesteps set up as above, and the eigenvalues of the model operator matrix M. The eigenvalues here are all contained within the circle $|\lambda| < 1$ and hence the model is stable in the discrete sense.

We now consider this example as a data assimilation problem, and compare the solutions obtained both by solving the saddle point formulation (3.17) using GM-

RES, and the low-rank approximation using LR-GMRES. For GMRES we also use a tolerance of 10^{-6}, and a maximum iteration number of 20. We take an assimilation window of 200 timesteps (giving $N = 199$) where observations are taken at each of these timesteps, followed by a forecast of 800 timesteps. Thus the resulting linear system (3.17) we solve here is of size $(200,000 + 200p)$, where p is the number of observations we take at each timestep. Independent of p, the full-rank update is $\delta x \in \mathbb{R}^{100,000}$. In contrast, the low-rank update is WV^T, where $W \in \mathbb{R}^{500\times r}, V \in \mathbb{R}^{200\times r}$. For $r = 20$, this requires only 14% of the storage of the full-rank update.

In the examples to follow, we compare the forecasts obtained after applying full- and low-rank solutions to the data assimilation problem with the forecast obtained from evolving the background estimate forward.

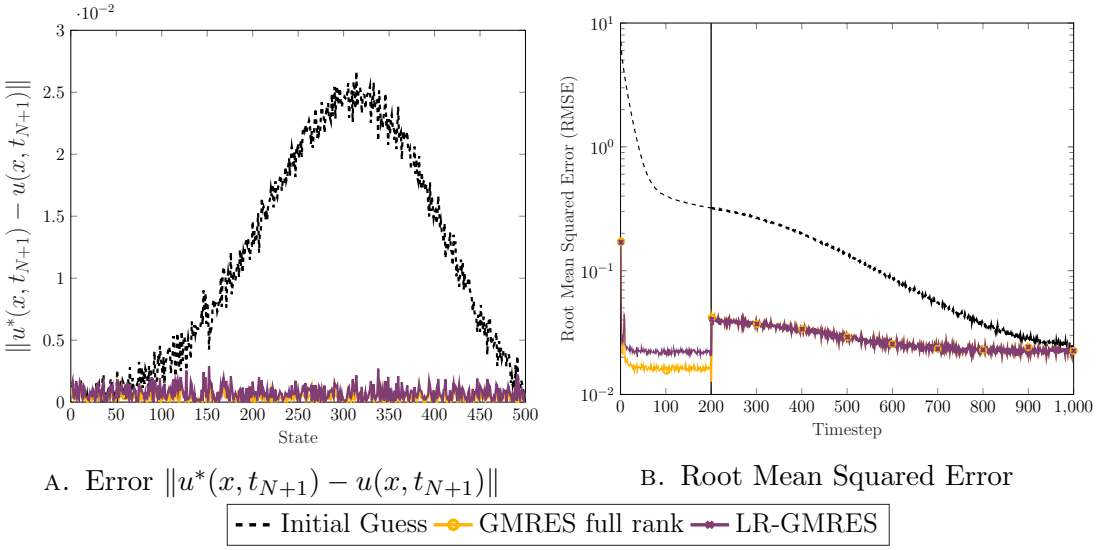
Perfect observations

First let us suppose we have perfect and full observations taken at every timestep in the assimilation window. Hence $p = 500$, and the size of the saddle point system we consider is $300,000$. We take as the background estimate u_0^b, a perturbed initial condition with background covariance $B = 0.1I_{500}$, and for this, and the following examples, we consider a model error with zero mean and covariance $Q = 10^{-6}I_{500}$.

Here, we take $r = 20$, which as described above requires only 14% of the storage used in the full-rank vector. Figure 3.2 a) shows the absolute error $\|u^*(x, t_{N+1}) - u(x, t_{N+1})\|$ for the time t_{N+1} at the end of the assimilation window, denoting the true solution by u^* computed by the numerical method, and in Figure 3.2 b) we consider the root mean squared error of the forecasts compared to the true state. In this example we see that the forecast obtained using the low-rank solver closely matches the one obtained from using GMRES despite the large reduction in space needed. During the assimilation window the low-rank approach results in a slightly higher RMSE than the full-rank method, but performs significantly more effectively than not applying data assimilation.

Partial, noisy observations

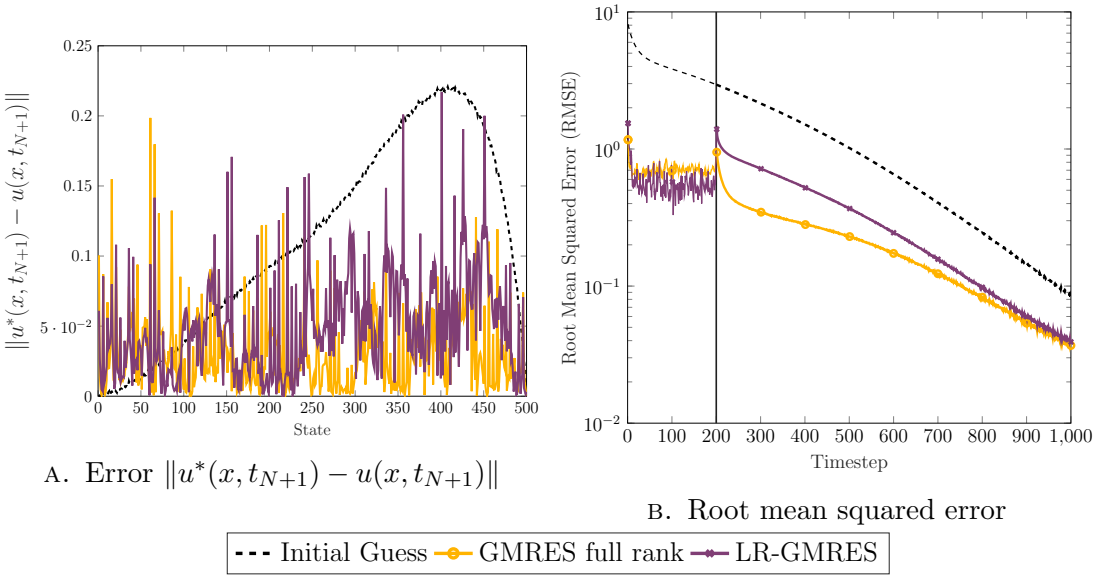
Let us now consider partial noisy observations, taking observations in every fifth component of u. These are generated from the truth with covariance $R = 0.01I_p$, for $p = 100$, and as such the linear system we consider for this example is of size $220,000$. In this example we take for the background error covariance $B_{i,j} = 0.1\exp(\frac{-|i-j|}{2n})$, keeping $Q = 10^{-6}I_{100}$ and $r = 20$. The resulting errors are shown in Figure 3.3.



A. Error $\|u^*(x, t_{N+1}) - u(x, t_{N+1})\|$ B. Root Mean Squared Error

Figure 3.2: Error at time t_{N+1}, and root mean squared error for the 1D advection-diffusion example with perfect observations ($r = 20$).

Here we see that the error between the true state and those obtained with the full- and low-rank data assimilation approaches are of similar levels for both approaches. When we consider the root mean squared errors of the full- and low-rank approaches in Figure 3.3 B) there is a difference between the resulting forecasts in contrast to Figure 3.2 B). The low-rank approach results in a forecast which has slightly higher levels of RMSE than the full-rank approach, but the error is still smaller than forecasting without applying data assimilation.



A. Error $\|u^*(x, t_{N+1}) - u(x, t_{N+1})\|$ B. Root mean squared error

Figure 3.3: Error at time t_{N+1}, and root mean squared error for the 1D advection-diffusion example with partial, noisy observations ($r = 20$).

Different choices of rank

Let us now consider the effect of the chosen rank on the assimilation result. In the previous examples we have considered $r = 20$, which resulted in the low-rank approximation to δx requiring only 14% of the storage needed for the full-rank solution. Here we consider $r = 5$ (requiring 3.5% of the storage), and $r = 1$ (needing just 0.7%), and otherwise keep the setup of the example used in Figure 3.3, with partial, noisy observations unchanged.

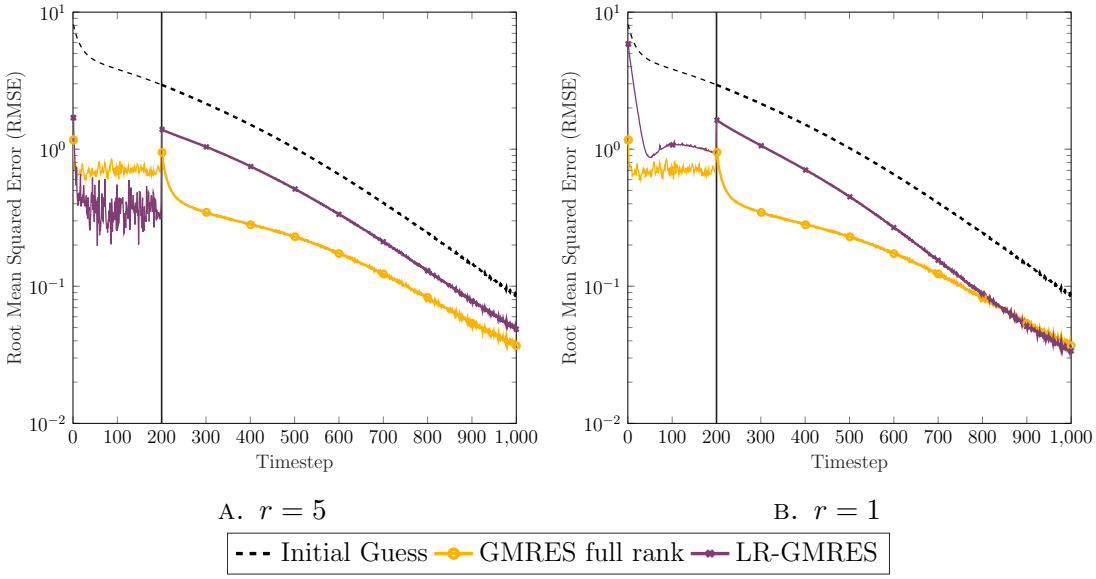


Figure 3.4: Root mean squared errors for the 1D advection-diffusion example with partial, noisy observations ($r = 5$, $r = 1$).

In Figure 3.4 A) we see that the forecast obtained from the low-rank method using $r = 5$, results in only a slightly larger level of error than that which we saw for $r = 20$ in Figure 3.3 B). In the assimilation window, taking $r = 5$ has a greater variability in the error, and indeed has a lower level than the forecast obtained with the full-rank method.

Surprisingly taking $r = 1$ results in a forecast which performs very similarly to the other two examples, despite having a larger error in the assimilation window than the full-rank example, and initially has a slightly higher level of error than the $r = 5$ case in the forecast window.

For this example, we see that the forecasts for both $r = 5$ and $r = 1$ are close to the full-rank solution and have a smaller error than not applying a data assimilation method.

Storage requirements

Table 3.1 presents the storage requirements for the examples considered in this section. As Figures 3.2- 3.4 demonstrate, despite the large reduction in the necessary storage for the low-rank approach, it results in close approximations to the full-rank method.

n	N	p	rank	# of matrix elements in full-rank solution	# of matrix elements in low-rank solution	storage reduction
100	199	100	20	20,000	6,000	70%
500	199	500	20	100,000	14,000	86%
500	199	100	20	100,000	14,000	86%
500	199	100	5	100,000	3,500	96.5%
500	199	100	1	100,000	700	99.3%

Table 3.1: Storage requirements for full- and low-rank methods in the 1D advection-diffusion equation examples.

Computation time

In Table 3.2, we present a comparison of the computation time for different choices of rank in the advection-diffusion example using LR-GMRES. For the following table, we consider the advection-diffusion example used in Figure 3.3, taking $n = 500, N = 199, p = 100$ leading to a saddle point matrix of size $220,000$. With each solver, we apply here only 20 iterations, and average over one hundred runs. These computations were done on an Intel i5-4460 processor operating at 3.2GHz.

Solver	runtime (s)
GMRES	9.0055
LR-GMRES (rank 50)	12.9397
LR-GMRES (rank 20)	2.5673
LR-GMRES (rank 5)	0.5909
LR-GMRES (rank 1)	0.3127

Table 3.2: Comparison of computation time for low-rank GMRES for the 1D advection-diffusion equation example.

We note that due to the truncation steps in the LR-GMRES algorithm, which are currently performed using a (sparse) `svd`, we do not see significant savings in Table 3.2 for the computation time for the larger choices of rank compared to

solving the saddle point system using ordinary GMRES because of this expense. However we see that in these examples, the small choice of rank still leads to close approximations to those obtained with GMRES. Furthermore, as seen in Table 3.1, these approximations require significantly lower storage requirements.

3.3.2 | Two-dimensional linearised shallow water equations

As a second example we consider the two-dimensional linearised shallow water equations (SWE), with a constant phase velocity. This example has two velocity components $u(x, y, t)$ and $v(x, y, t)$ and a height perturbation $\eta(x, y, t)$, where $(x, y) \in [0, 1] \times [0, 1]$ is a spatial coordinate and $t > 0$ is time. The governing PDEs are:

$$\frac{\partial u}{\partial t} = -\frac{\partial \eta}{\partial x}, \qquad \frac{\partial u}{\partial t} = -\frac{\partial \eta}{\partial y}, \qquad \frac{\partial \eta}{\partial t} = -\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right),$$

with the initial conditions

$$u(x, y, 0) = 0, \qquad v(x, y, 0) = 0, \qquad \eta(x, y, 0) = \eta_0(x, y),$$

where $\eta_0(x, y)$ is a sinusoidal perturbation.

We solve this problem using centered finite differences, discretising the space with an $m \times m$ grid taking $m = 13$, thus leading to a state space size of $n = 507$ considering the height and two velocities, and taking timesteps of size $\Delta t = 5 \cdot 10^{-4}$.

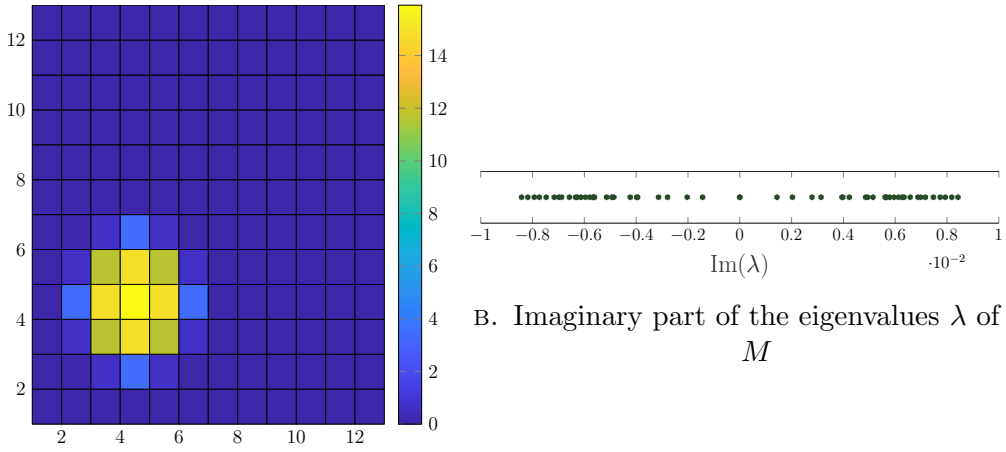
As with the advection-diffusion example when considering this as a data assimilation problem, we take an assimilation window of $(N + 1) = 200$ timesteps with observations taken at each of these timesteps, followed by a forecast of 800 timesteps.

In Figure 3.5 we see the initial condition $\eta_0(x, y)$ for this example as set up above, and the imaginary part of the eigenvalues of the model operator M. The eigenvalues λ for this example are of the form $1 + \nu i$, with $\nu \in (-0.01, 0.01)$. Since this results in some $|\lambda| > 1$, the model is not stable.

For the following numerical examples, we consider the RMSE for just the height component of the state. This is for convenience and to present clearer figures as the velocity components behave similarly.

Perfect observations

As in the advection-diffusion example, let us first suppose we have perfect observations taken at every state in the assimilation window. Hence $p = 507$, and the

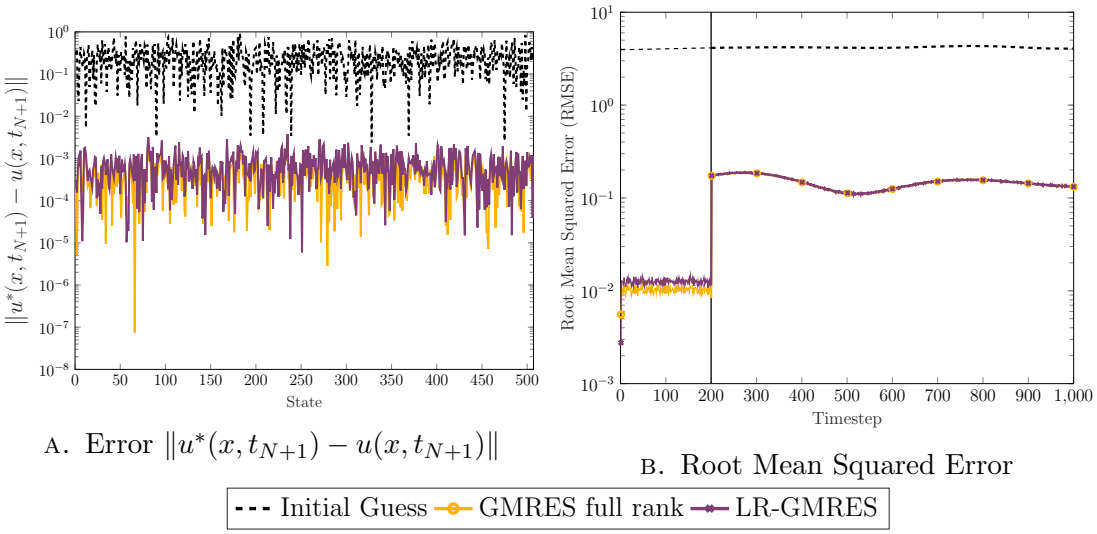


A. Shallow water equations initial condition $\eta_0(x, y)$

B. Imaginary part of the eigenvalues λ of M

Figure 3.5: The initial condition for the 2D shallow water equations example and the eigenvalues of the model operator M

size of the saddle point system we consider is $304,200$. We take as the background estimate u_0^b, a perturbed initial condition with background covariance $B = 0.1I_{507}$, and for this, and the following examples, we consider a model error with zero mean and covariance $Q = 10^{-6}I_{507}$.



A. Error $\|u^*(x, t_{N+1}) - u(x, t_{N+1})\|$

B. Root Mean Squared Error

Figure 3.6: Error at time t_{N+1}, and root mean squared error for the 2D shallow water equations example with perfect observations ($r = 20$).

Here, we take $r = 20$ and observe in Figure 3.6 that, as with our perfect observations example for the advection-diffusion problem, the forecast obtained using the low-rank solver achieves very similar levels of error to those obtained via GMRES, despite once again using only 14% of the storage for the update vector.

Partial, noisy observations

If we now consider partial noisy observations, taking $p = 100$ with observations in every fifth component in our state. These are generated from the truth with covariance $R = 0.01I_{100}$, and as such the linear system we consider for this example is of size $222,800$. In this example we use $B_{i,j} = 0.1\exp(\frac{-|i-j|}{2n})$ and $Q = 10^{-6}I_{100}$. The resulting root mean squared errors for the forecasts are shown in Figure 3.7 for $r = 20$ and $r = 5$.

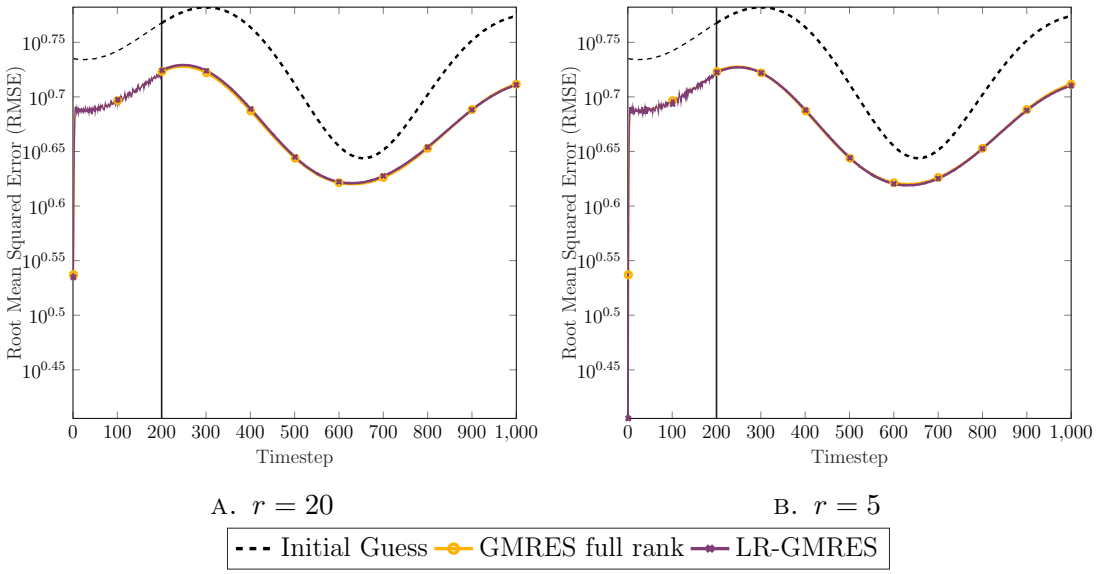


Figure 3.7: Root mean squared errors for the 2D shallow water equations example with partial, noisy observations ($r = 20$, $r = 5$).

We observe in Figure 3.7 that with partial observations for this example we obtain near identical levels of error for the forecasts obtained through both GMRES and LR-GMRES, irrespective of the choice of rank for these examples, with low-rank updates requiring much less storage.

Storage requirements

Table 3.3 presents the storage requirements for the examples considered in this section. As for the previous example, despite the large reduction in the necessary storage for the low-rank approach, it results in close approximations to the full-rank method.

n	N	p	rank	# of matrix elements in full-rank solution	# of matrix elements in low-rank solution	storage reduction
507	199	507	20	101,400	14,140	86%
507	199	100	20	101,400	14,140	86%
507	199	100	5	101,400	3,535	96.5%

Table 3.3: Storage requirements for full- and low-rank methods in the 2D shallow water equations examples.

3.4 | Time-dependent systems

Let us now consider an extension of the Kronecker formulation (3.19) to the time-dependent case, allowing for time-dependent model, and observation operators, and the respective covariance matrices.

3.4.1 | Kronecker formulation of time-dependent systems

The remaining assumption we must make is that the number of observations in the i-th timestep, p_i is constant, i.e. $p_i = p$ for each i. With these assumptions, the linear system in (3.19) becomes

$$\begin{bmatrix} F_1 \otimes B + \sum_{i=1}^{N} F_{i+1} \otimes Q_i & 0 & I \otimes I_x + \sum_{i=1}^{N} C_i \otimes M_i \\ 0 & \sum_{i=0}^{N} F_{i+1} \otimes R_i & \sum_{i=0}^{N} F_{i+1} \otimes H_i \\ I \otimes I_x + \sum_{i=1}^{N} C_i^T \otimes M_i^T & \sum_{i=0}^{N} F_{i+1} \otimes H_i^T & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \mu \\ \delta x \end{bmatrix} = \begin{bmatrix} b \\ d \\ 0 \end{bmatrix}, \tag{3.45}$$

where F_i denotes the matrix with 1 on the ith entry of the diagonal, and zeros elsewhere, and C_i is the matrix with -1 on the ith column of the subdiagonal, and zeros elsewhere. Here M_i and H_i are linearisations of the model and observation operators $\mathcal{M}_i$ and $\mathcal{H}_i$ respectively about x_i.

As in Section 3.2.1, we may use (3.18) to rewrite this as the following (now more

general) matrix equations

$$\begin{aligned} B\Lambda F_1 + \sum_{i=1}^{N} Q_i \Lambda F_{i+1} + X + \sum_{i=1}^{N} M_i X C_i^T &= \mathbb{b}, \\ \sum_{i=0}^{N} R_i U F_{i+1} + \sum_{i=0}^{N} H_i X F_{i+1} &= \mathbb{d}, \\ \Lambda + \sum_{i=1}^{N} M_i^T \Lambda C_i + \sum_{i=0}^{N} H_i^T U F_{i+1} &= 0. \end{aligned} \tag{3.46}$$

Here as before, $\lambda, \delta x, b, \mu$ and d are vectorised forms of the matrices $\Lambda, X, \mathbb{b} \in \mathbb{R}^{n\times N+1}$ and $U, \mathbb{d} \in \mathbb{R}^{p\times N+1}$ respectively. These matrix equations must again be solved for Λ, U and X, where X is the matrix of interest.

Algorithm 3 is an implementation of `Amult` for the time-dependent case, explicitly writing the concatenation defined by (3.46) in the form required for LR-GMRES. This requires linearisations of the model and observation operators at all timesteps in order to be applied.

Algorithm 3: Matrix multiplication (time-dependent) (`Amult`)

Input: $W_{11}, W_{12}, W_{21}, W_{22}, W_{31}, W_{32}$
Output: $Z_{11}, Z_{12}, Z_{21}, Z_{22}, Z_{31}, Z_{32}$

$$\begin{aligned} Z_{11} &= [BW_{11}, \quad Q_1W_{11}, \quad \ldots, \quad Q_NW_{11}, \quad W_{31}, \quad M_1W_{31}, \quad \ldots, \quad M_NW_{31}], \\ Z_{12} &= [F_1W_{12}, \quad F_2W_{12}, \quad \ldots, \quad F_{N+1}W_{12}, \quad W_{32}, \quad C_1W_{32}, \quad \ldots, \quad C_NW_{32}], \\ Z_{21} &= [R_0W_{21}, \quad \ldots, \quad R_NW_{21}, \quad H_0W_{31}, \quad \ldots, \quad H_NW_{31}], \\ Z_{21} &= [F_1W_{22}, \quad \ldots, \quad F_{N+1}W_{22}, \quad F_1W_{32}, \quad \ldots, \quad F_{N+1}W_{32}], \\ Z_{31} &= [W_{11}, \quad M_1^TW_{11}, \quad \ldots, \quad M_N^TW_{11}, \quad H_0^TW_{21}, \quad \ldots, \quad H_N^TW_{21}], \\ Z_{32} &= [W_{12}, \quad C_1^TW_{12}, \quad \ldots, \quad C_N^TW_{12}, \quad F_1W_{22}, \quad \ldots, \quad F_{N+1}W_{22}] \end{aligned}$$

We note that further to the truncation expense highlighted in Section 3.3, the significantly increased number of matrices being concatenated prior to truncation results in longer runtimes, particularly if new linearised matrices must be computed.

As an example, we consider the Lorenz-95 system [94] which is both nonlinear, and also chaotic rather than smoothing such as the previous example (Section 3.3.1), so as to better represent real world data assimilation problems such as weather forecasting.

3.4.2 | Lorenz-95 system

We consider the Lorenz-95 system [94], this is a generalisation of the three dimensional Lorenz system [93] to n dimensions. The model is defined by a system of n

nonlinear ordinary differential equations

$$\frac{\mathrm{d}z^i}{\mathrm{d}t} = -z^{i-2}z^{i-1} + z^{i-1}z^{i+1} - z^i + f, \tag{3.47}$$

where $z = [z^1, z^2, \ldots, z^n]^T$ is the state of the system, and f is a forcing term. It is known that for $f = 8$, the Lorenz system exhibits chaotic behaviour [57, 94]. Also noted is that for reasonably large values of n (here we take $n = 40$), this choice of f leads to a model which is comparable to weather forecasting models.

We solve (3.47) using a 4th order Runge-Kutta method in order to obtain

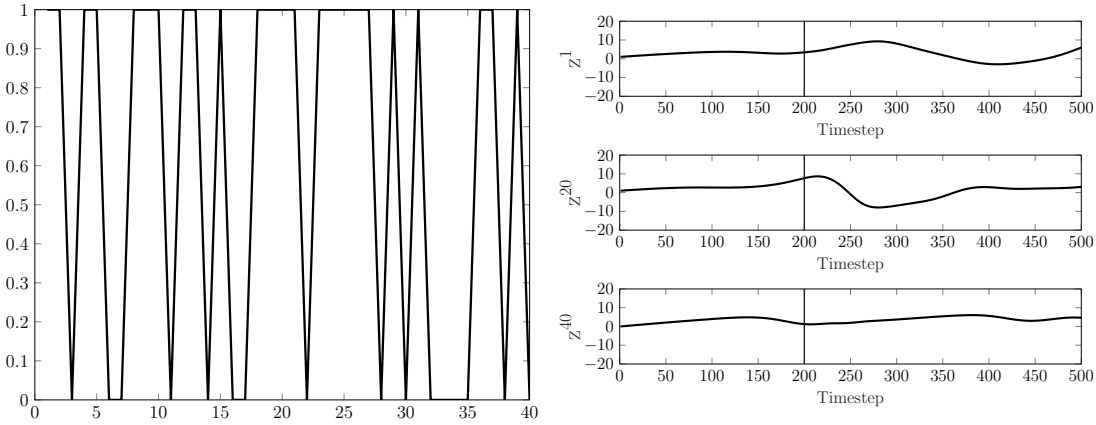
$$z_{k+1} = \mathcal{M}_k(z_k), \quad \text{where } z_k = [z_k^1, z_k^2, \ldots, z_k^n]^T, \tag{3.48}$$

where $\mathcal{M}_k$ is the nonlinear model operator which evolves the state z_k to z_{k+1}. As before $\mathcal{H}_k$ denotes the potentially nonlinear observation operator for the state z_k. We set the initial value of each z^i to be "1" or "0" with equal probability.

To formulate the data assimilation problem as a saddle point problem, we generate the tangent linear model, and observation operators M_k and H_k by linearising $\mathcal{M}_k$ and $\mathcal{H}_k$ about z_k.

As in Section 3.3.1, we compare the low-rank approximation computed using LR-GMRES, to the full-rank solution of the saddle point formulation (3.17) solved using GMRES, and the background estimate (e.g. no assimilation). We perform the data assimilation using an assimilation window of 200 timesteps, where observations are taken at each of these timesteps, followed by a forecast of 800 timesteps, all of size $\Delta t = 5 \cdot 10^{-3}$. The resulting full-rank update for the 40-dimensional Lorenz system is therefore $\delta x \in \mathbb{R}^{8,000}$, whilst in contrast the low-rank update WV^T, is such that $W \in \mathbb{R}^{40\times r}, V \in \mathbb{R}^{200\times r}$. Here we consider $r = 20$ once more, which here requires 60% of the storage, still demonstrating a significant reduction compared to the full-rank GMRES solve.

In Figure 3.8 we see the initial condition for this example as described above. Additionally we consider the evolution of the components z^1, z^{20} and z^{40} over the forecast and assimilation window, and observe that these states behave very differently to one another.



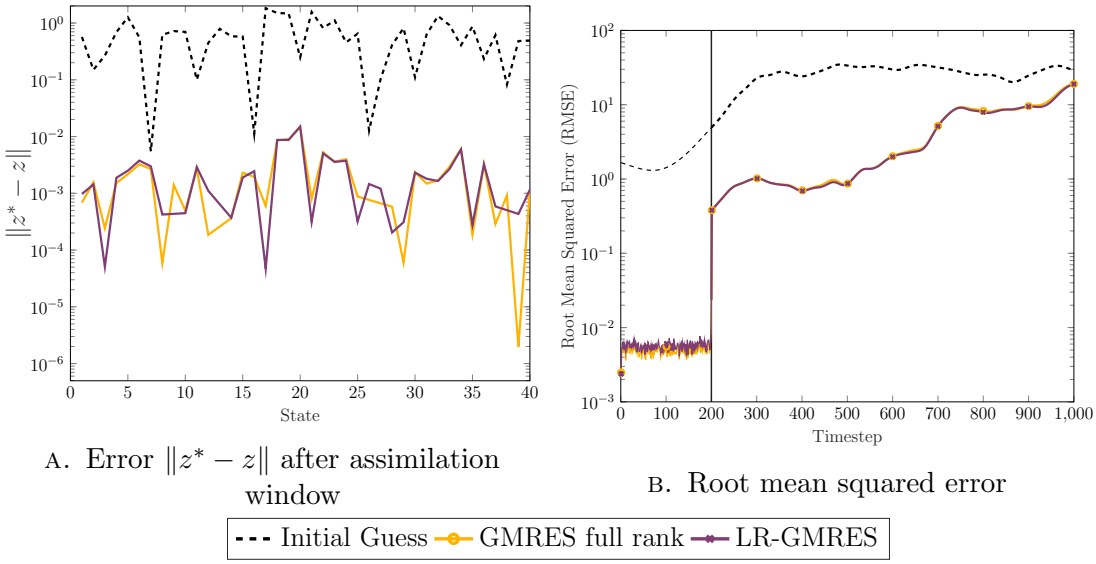
A. Lorenz-95 initial condition

B. Forecast of z^1, z^{20} and z^{40}

Figure 3.8: The initial condition for the 40-dimensional Lorenz-95 example and the evolution of three components for 1000 timesteps.

Perfect observations

As with the advection-diffusion equation, let us first suppose we have perfect observations of every state in the assimilation window, we take as the background estimate x_0^b, a perturbation of the "1,0" initial condition with background covariance $B = 0.1I_{40}$, and as before, we consider a model error with covariance $Q = 10^{-4}I_{40}$. The error $\|z^* - z\|$ between the true state z^*, and the assimilated state z, for the timestep t_{N+1} immediately after the assimilation window, and the root mean square errors for the three approaches in this example are presented in Figure 3.9.



A. Error $\|z^* - z\|$ after assimilation window

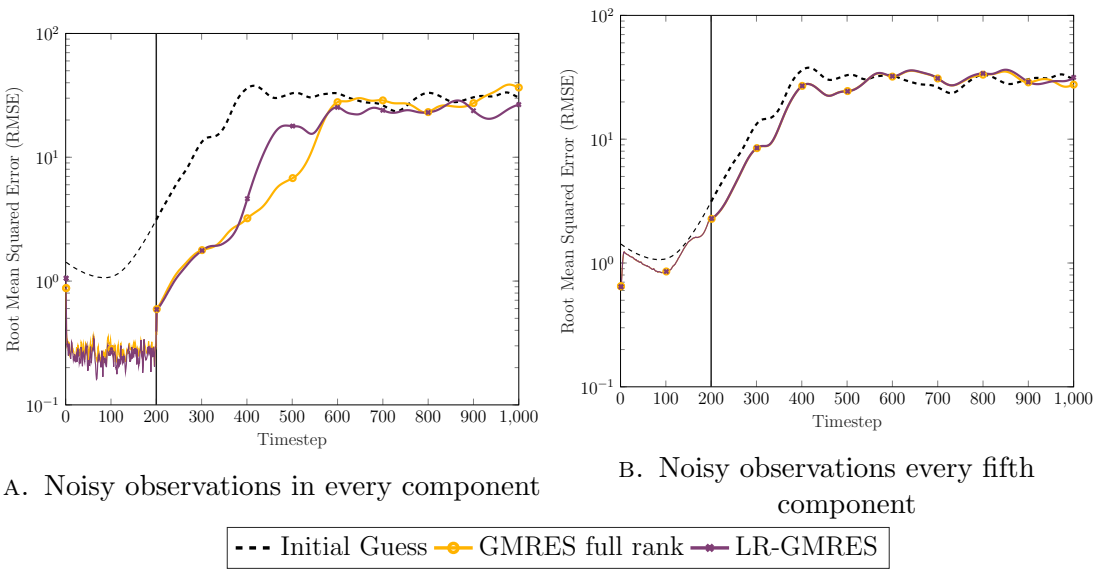
B. Root mean squared error

Figure 3.9: Error at time t_{N+1}, and root mean squared error for the 40-dimensional Lorenz-95 system with perfect observations ($r = 20$).

In this example with perfect observations, as with the advection-diffusion example, we see that the forecast obtained using LR-GMRES has a very similar level of error throughout the window considered, to that obtained using ordinary GMRES, with a solution which requires 40% less storage. In the state error plot we observe small differences between the approaches for some states, however is still very similar.

Noisy observations

We next consider noisy observations, taking $R = 0.01I_p$ for the observation error covariance, and as the background error covariance $B_{i,j} = 0.1\exp(\frac{-|i-j|}{2n})$. In Figure 3.10 we consider the root mean squared errors for two different choices of observation operator: taking interpolatory observations in every component ($p = 40$) shown on the left, and in every fifth component ($p = 8$) on the right. In both cases,



A. Noisy observations in every component

B. Noisy observations every fifth component

Figure 3.10: Root mean squared error for the 40-dimensional Lorenz-95 system with noisy, and partial observations ($r = 5$, $r = 1$).

we see the forecast generated from the low-rank method matches that from the full-rank very closely until timestep 400 in Figure 3.10 A), and throughout for Figure 3.10 B). To achieve these very similar results using the low-rank approach, despite using just 60% of the storage, is very promising.

500-dimensional Lorenz-95

Finally, we consider as a larger example, the 500 - dimensional Lorenz-95 system with an assimilation window of 200 timesteps. This gives a full-rank update $\delta x \in$

$\mathbb{R}^{100,000}$, and we consider two different choices of low-rank, $r = 20$ requiring 14% of the storage, and $r = 5$ needing 3.5%. In this example we take noisy observations in each state, with covariances $B_{i,j} = 0.1 \exp(\frac{-|i-j|}{2n})$, $R = 0.01 I_{500}$ and $Q = 10^{-6} I_{500}$.

These examples, shown in Figure 3.11 demonstrate further that a low-rank approximation performs very closely to that of the full-rank solution despite taking a smaller rank r.

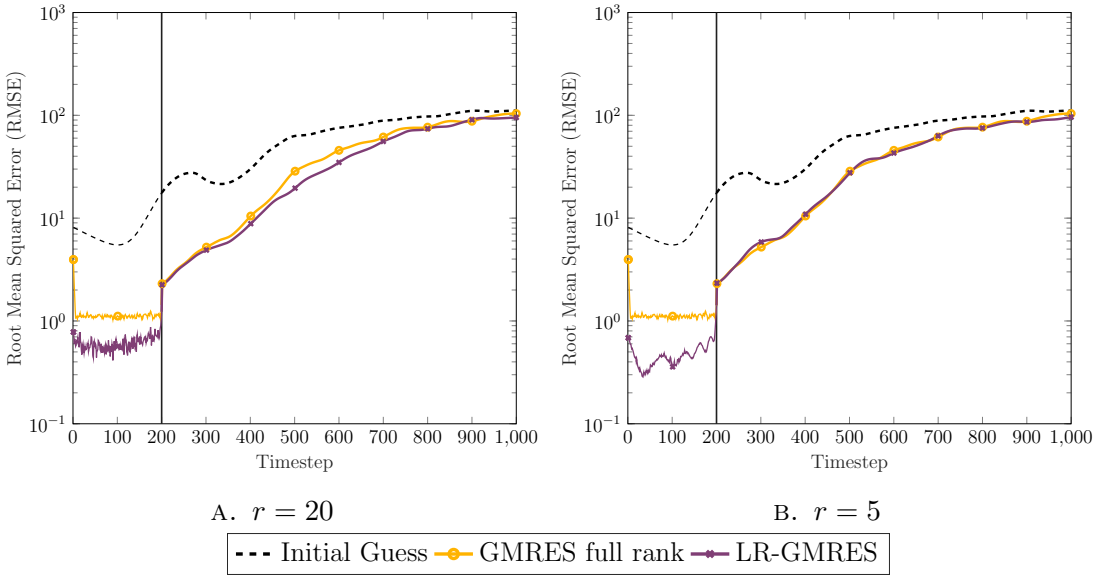


Figure 3.11: Root mean squared error for the 500-dimensional Lorenz-95 system with full, noisy observations ($r = 20$, $r = 5$).

During the forecast window for both $r = 20$ and $r = 5$, we see the forecast obtained from the low-rank solution results in very similar levels of RMSE for this example, despite the large reduction in size.

Table 3.4 presents the storage requirements for the examples considered in this section. As with the other two examples, despite the large reduction in storage required, the experiments have shown that the low-rank approximations give similar results to the full-rank approach, which is a very good prospect.

n	N	p	rank	# of matrix elements in full-rank solution	# of matrix elements in low-rank solution	storage reduction
40	199	40	20	8,000	4,800	40%
40	199	8	20	8,000	4,800	40%
500	199	500	20	100,000	14,000	86%
500	199	500	5	100,000	3,500	96.5%

Table 3.4: Storage requirements for full- and low-rank methods in the Lorenz-95 examples.

3.5 | Conclusions

The saddle point formulation of weak constraint four-dimensional variational data assimilation results in a large linear system which in the incremental approach is solved to determine the update δx at every step. In this chapter we have proposed a low-rank approach which approximates the solution to the saddle point system, with significant reductions in the storage needed. This was achieved by considering the structure of this saddle point system and using techniques from the theory of matrix equations. Using the properties of the Kronecker product we showed that low-rank solutions to the data assimilation problem exist under certain assumptions, with numerical experimentation demonstrating that this may be the case even when these assumptions are relaxed.

We introduced a low-rank GMRES solver and considered the requirements for implementing this algorithm. Numerical experiments have demonstrated that the low-rank approach introduced here is successful using both linear and nonlinear models.

In these examples we achieved close approximations to the full-rank solutions with storage requirements as low as 1% of those needed by the full-rank approach, and can be obtained in less time than through GMRES. These results are very promising, though some further investigation is needed, in particular for nonlinear problems.

In the next chapter, we consider preconditioning approaches for the data assimilation saddle point problem, and the difficulties which arise when applying preconditioners to the low-rank method introduced here.

CHAPTER 4

PRECONDITIONING THE DATA ASSIMILATION SADDLE POINT PROBLEM

This chapter considers preconditioning the data assimilation saddle point problem from Chapter 3. In [55] preconditioning with LR-GMRES was considered, and this chapter extends this investigation, providing context with the traditional saddle point problem.

4.1 | Introduction

When solving a linear system $\mathcal{A}x = b$ iteratively using for example a Krylov subspace method such as MINRES [98] or GMRES [111], convergence is usually slow.

For symmetric problems, or more generally when the matrix is normal, the (worst case) convergence behaviour of Krylov subspace methods such as MINRES and GMRES is completely determined by its spectrum. In the nonnormal case, the analysis of the convergence of GMRES is more complicated and may not be related to the eigenvalues [88].

To illustrate this, we consider MINRES, noting that GMRES and MINRES are theoretically equivalent in exact arithmetic for symmetric problems. The relative residual norm of this method can be written

$$\frac{\|r_k\|}{\|r_0\|} \leq \min_{p_k \in \prod_k} \max_{\lambda \in \sigma(\mathcal{A})} |p_k(\lambda)|. \tag{4.1}$$

where r_k denotes the residual after k iterations, $\prod_k$ is the set of degree k polynomials with $p_k(0) = 1$, and $\sigma(\mathcal{A})$ denotes the spectrum of $\mathcal{A}$.

We observe that this indicates that the iterative method will converge to the

solution after s iterations if $\mathcal{A}$ has s distinct eigenvalues, and thus the total number of iterations is at most the size of $\mathcal{A}$. For the data assimilation saddle point problem, this could be $(2n+p)(N+1)$ iterations. As noted in [65, 140], if the eigenvalues of $\mathcal{A}$ are in a small number of clusters, neither too far, nor too close to one side of the origin, the Krylov subspace method should converge rapidly.

If more information is known about the properties of the matrix $\mathcal{A}$ and/or its spectra, further convergence estimates can be computed. We refer to [65, 88] for more discussion on this topic, and convergence bounds for GMRES.

As a result of slow convergence, we often (implicitly) transform the system into one with more desirable properties so as to reduce the number of iterations needed to obtain a solution. This is particularly important for GMRES, as each iteration increases the storage requirements. A preconditioner is a matrix $\mathcal{P}$ which performs this transformation, and can be applied on the left of the system ($\mathcal{P}^{-1}\mathcal{A}x = \mathcal{P}^{-1}b$) or on the right ($\mathcal{A}\mathcal{P}^{-1}u = b, u = \mathcal{P}x$). It is also possible to precondition on both sides and consider split preconditioning ($\mathcal{P}_1^{-1}\mathcal{A}\mathcal{P}_2^{-1}u = \mathcal{P}_1^{-1}b, u = \mathcal{P}_2 x$). Whether to use left-, right- or split preconditioning is problem and solving method dependent. Here we consider right preconditioning for our problems because the residuals for the right-preconditioned system are identical to the true residuals in exact arithmetic.

When choosing a preconditioner the aim is to improve the spectral properties of the resulting preconditioned system, either in terms of clustering or location of the eigenvalues or the spectral condition number of the matrix $\mathcal{A}\mathcal{P}^{-1}$ (or indeed $\mathcal{P}^{-1}\mathcal{A}$). As such, the matrix $\mathcal{P}$ often approximates $\mathcal{A}$ in some sense.

The art of choosing a preconditioner is a large area of research in numerical linear algebra. There are multiple different approaches to designing preconditioners, and one which works well for one problem may be ineffective for another. It is very problem dependent, and this is particularly true in saddle point problems as noted in [14]; one must exploit the block structure, and any knowledge of the origin or structure of the individual blocks in order to construct an effective preconditioner.

In this chapter we consider the preconditioning of the weak constraint data assimilation saddle point problem we introduced in Chapter 3, and how preconditioning strategies change when considering the low-rank method introduced there.

4.2 | Preconditioning the data assimilation saddle point problem

We return to the saddle point problem

$$\begin{bmatrix} \mathbf{D} & 0 & \mathbf{L} \\ 0 & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^T & \mathbf{H}^T & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \mu \\ \delta x \end{bmatrix} = \begin{bmatrix} b \\ d \\ 0 \end{bmatrix}, \tag{4.2}$$

from Chapter 3 and consider preconditioners for this problem. For the remainder of this chapter we shall refer to the saddle point matrix in (4.2) as $\mathcal{A}$.

General saddle point matrices $\mathbf{A}$ are often thought of as block 2×2 matrices of the form

$$\mathbf{A} = \begin{bmatrix} A_1 & B_1^T \\ B_2 & C_1 \end{bmatrix}, \tag{4.3}$$

where $A_1 \in \mathbb{R}^{n \times n}$, $B_1, B_2 \in \mathbb{R}^{m \times n}$, $C_1 \in \mathbb{R}^{m \times m}$ where $n \geq m$. For the saddle point problem (4.2) we consider the partitioning

$$A_1 = \begin{bmatrix} \mathbf{D} & 0 \\ 0 & \mathbf{R} \end{bmatrix}, \quad B_1 = B_2 = \begin{bmatrix} \mathbf{L}^T & \mathbf{H}^T \end{bmatrix}, \quad C_1 = 0.$$

These blocks are often referred to in the literature as the $(1,1)$, $(1,2)$ or $(2,1)$, and $(2,2)$ blocks respectively.

Many approaches exist for preconditioning saddle point problems, a number of which are detailed in [14, 15, 108]. However, the data assimilation setting introduces an unusual situation where the $(1,2)$ block $\begin{bmatrix} \mathbf{L} \\ \mathbf{H} \end{bmatrix}$ of the saddle point matrix is more computationally expensive than the $(1,1)$ block $\begin{bmatrix} \mathbf{D} & 0 \\ 0 & \mathbf{R} \end{bmatrix}$. This is the case because the linearised model and observation operators are present in the $(1,2)$ block, whilst the $(1,1)$ block consists of our covariance matrices which are comparatively easier to use. In other applications, the opposite is more typically the case, with the matrices B_2 (and B_1^T) in (4.3) arising as a constraint, whilst the matrix A_1 contains information about the model, which may be a discretisation of a differential operator, multiplication by a function, or a finite element mass matrix to give just a few examples.

When constructing preconditioners for this problem we therefore do not consider approximations to the matrices $\mathbf{D}$ or $\mathbf{R}$, however we shall use approximations to

$\mathbf{L} = I_{N+1} \otimes I_n + C \otimes M$ and $\mathbf{H} = I_{N+1} \otimes H$, namely $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{H}}$.

A natural choice of approximation $\tilde{\mathbf{L}}$ is one of the form

$$\tilde{\mathbf{L}} = I_{N+1} \otimes I_n + C \otimes \tilde{M}, \tag{4.4}$$

where C as before is the tridiagonal matrix with -1 on the subdiagonal and $\tilde{M}$ is an approximation to the linearised model operator M, thus retaining the structure of $\mathbf{L}$. We consider the simple approximation taking $\tilde{M} = I_n$ and introduce

$$\hat{\mathbf{L}} = I_{N+1} \otimes I_n + C \otimes I_n = (I_{N+1} + C) \otimes I_n. \tag{4.5}$$

We also consider the approximation $\tilde{\mathbf{L}} = I_{N+1} \otimes I_n$ which for convenience we denote $\mathbf{I}$ for the remainder of this chapter. In Section 4.6 we consider the implementation of more general approximations of the form (4.4).

An important tool used in saddle point preconditioners is the Schur complement of the $(1,1)$ block in the saddle point matrix: $\mathbf{S}$, an $(N+1)n \times (N+1)n$ matrix of the form

$$\mathbf{S} = -\mathbf{L}^T\mathbf{D}^{-1}\mathbf{L} - \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}, \tag{4.6}$$

for the saddle point problem (4.2) which has the inverse

$$\mathbf{S}^{-1} = -\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T} + \mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}\mathbf{H}^T(\mathbf{R} + \mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}\mathbf{H}^T)^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}. \tag{4.7}$$

As with the matrices $\mathbf{L}$ and $\mathbf{H}$, when using the Schur complement in practice, we must consider approximations. One can approximate the Schur complement separately from the other terms however here we use the approximations $\tilde{\mathbf{S}}$ arising from the approximation of $\mathbf{L}$ and $\mathbf{H}$.

In Table 4.1 we present the approximations to $\mathbf{L}$, $\mathbf{H}$ and $\mathbf{S}$ which we use in this chapter.

$\tilde{\mathbf{L}}$	$\tilde{\mathbf{H}}$	$\tilde{\mathbf{S}}$	$\tilde{\mathbf{S}}^{-1}$
$\mathbf{I}$	0	$-\mathbf{D}^{-1}$	$-\mathbf{D}$
$\mathbf{I}$	$\mathbf{H}$	$-\mathbf{D}^{-1} - \mathbf{H}\mathbf{R}^{-1}\mathbf{H}$	$-\mathbf{D} + \mathbf{D}\mathbf{H}^T(\mathbf{R} + \mathbf{H}\mathbf{D}\mathbf{H}^T)^{-1}\mathbf{H}\mathbf{D}$
$\hat{\mathbf{L}} = (I_{N+1} + C) \otimes I_n$	0	$-\hat{\mathbf{L}}^T\mathbf{D}^{-1}\hat{\mathbf{L}}$	$-\hat{\mathbf{L}}^{-1}\mathbf{D}\hat{\mathbf{L}}^{-T}$

Table 4.1: Table of approximations for $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{H}}$ and the resulting Schur complement approximations.

Whilst we do not consider it here, in Section 4.6 we present approximations to the Schur complement inverse using approximations to $\mathbf{L}^{-1}$.

4.2.1 | Spectral properties of the data assimilation saddle point problem

As we described in Section 4.1, by preconditioning we aim to improve the spectral properties of the resulting preconditioned system, either in terms of clustering of the eigenvalues or the spectral condition number of the matrix $\mathcal{A}\mathcal{P}^{-1}$. In this section, we shall consider the spectral properties of the saddle point matrix $\mathcal{A}$ prior to preconditioning.

We make use of the following result from [109] which provides eigenvalue bounds for a class of saddle point systems, such as the data assimilation saddle point problem.

Theorem 4.1. *[109] Let* $\mathbf{A}$ *be the matrix*

$$\mathbf{A} = \begin{bmatrix} A_1 & B_1^T \\ B_1 & 0 \end{bmatrix},$$

with $A_1 \in \mathbb{R}^{n\times n}$ *symmetric positive definite, and* $B_1 \in \mathbb{R}^{n\times m}, m \leq n$ *of full rank. We denote the largest and smallest eigenvalues of* A_1 μ_1 *and* μ_n *respectively, and* σ_1, σ_m *the largest and smallest singular values of* B_1*. Let* $\sigma(\mathbf{A})$ *be the spectrum of* $\mathbf{A}$*. Then*

$$\sigma(\mathbf{A}) \subset I^- \cup I^+,$$

where

$$I^- = \left[\frac{1}{2}\left(\mu_n - \sqrt{\mu_n^2 + 4\sigma_1^2}\right), \frac{1}{2}\left(\mu_1 - \sqrt{\mu_1^2 + 4\sigma_m^2}\right)\right],$$

and

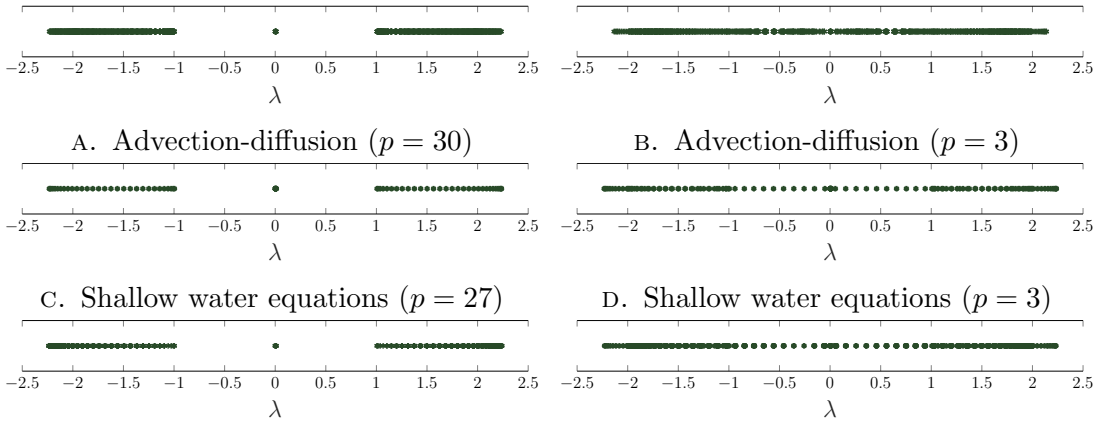
$$I^+ = \left[\mu_n, \frac{1}{2}\left(\mu_1 + \sqrt{\mu_1^2 + 4\sigma_1^2}\right)\right].$$

We apply the above result for the data assimilation saddle point matrix.

As in Chapter 3, we consider three different model problems: the 1D advection-diffusion example, the linearised 2D shallow water equations, and the Lorenz-95 problem. For these illustrative examples we consider an assimilation window of $30 = N+1$ timesteps for all three problems, with observations at each of these timesteps. A state space discretisation is taken with $n = 30$ for the advection-diffusion and Lorenz-95 problems, and $n = 27$ for the shallow water equations example. We consider both full observations, i.e. $p = n$, and partial observations with $p = 3$. In both scenarios we assume the observation errors have zero mean with covariance $R = 0.01I_p$ and for the background and model errors we have a zero mean and covariances $B = 0.01I_n$ and $Q = 10^{-4}I_n$ respectively. With the covariance matrices

thus defined, when using the result from Theorem 4.1, the largest eigenvalue arising from the covariance matrices is $\mu_1 = 0.01$, with the smallest being $\mu_n = 0.0001$. For the nonlinear Lorenz-95 example, we consider the linearisation arising in the first inner loop of incremental 4D-Var.

Let us now consider the eigenvalues of the resulting saddle point matrix $\mathcal{A}$ for these three problems with full $p = n$ observations, and partial $p = 3$ observations for the three different models. We plot the eigenvalues in Figure 4.1, and we present in Table 4.2 the resulting bounds from Theorem 4.1.



a. Advection-diffusion ($p = 30$)

b. Advection-diffusion ($p = 3$)

c. Shallow water equations ($p = 27$)

d. Shallow water equations ($p = 3$)

e. Lorenz ($p = 30$)

f. Lorenz ($p = 3$)

Figure 4.1: Eigenvalues of $\mathcal{A}$ with different model operators.

We observe that the eigenstructures are similar across the three different choices of model operator. The difference in the clustering is affected by the model operator, with the advection-diffusion example resulting in a far greater number of (numerically) distinct eigenvalues.

In contrast, reducing the number of observations reduces the clustering, with the eigenvalues closer to zero which is highlighted in Table 4.2. Here we use the notation AD and SWE to refer to the advection-diffusion and shallow water equations problems respectively.

Whilst Theorem 4.1 does not distinguish between the positive eigenvalues clustered near 0, and those between 1 and 2.5 in Figures 4.1 a), 4.1 c) and 4.1 e) for the positive interval I^+, the greater spread is apparent in the interval I^-.

As noted above, for all of these examples, the largest eigenvalue of $\begin{bmatrix} \mathbf{D} & 0 \\ 0 & \mathbf{R} \end{bmatrix}$ is $\mu_1 = 0.01$, with the smallest being $\mu_n = 0.0001$.

As we shall see in the subsequent sections, whilst the choice of model does not play a large role on the eigenstructure of the matrix $\mathcal{A}$, it has a greater effect when considering preconditioned systems, and the efficacy of those preconditioners when

Model	σ_1	σ_m	I^-	I^+
AD ($p = 30$)	2.2329	1.0014	[-2.2329, -0.9964]	[0.0001, 2.2379]
SWE ($p = 27$)	2.2332	1.0016	[-2.2331, -0.9966]	[0.0001, 2.2382]
Lorenz ($p = 30$)	2.2380	1.0012	[-2.2379, -0.9962]	[0.0001, 2.2430]
AD ($p = 3$)	2.1364	0.0567	[-2.1364, -0.0519]	[0.0001, 2.1415]
SWE ($p = 3$)	2.2337	0.0515	[-2.2336, -0.0467]	[0.0001, 2.2387]
Lorenz ($p = 3$)	2.2295	0.0494	[-2.2295, -0.0446]	[0.0001, 2.2345]

Table 4.2: Extreme singular values of $[\mathbf{L}^T \quad \mathbf{H}^T]$, and eigenvalue bounds for $\mathcal{A}$ with different model operators.

used in GMRES due to the clustering observed in Figure 4.1.

In the following sections we consider applying preconditioners using the approximations in Table 4.1. We investigate two classes of preconditioner which are commonly used for saddle point problems and exploit the block structure: Schur complement preconditioners, and constraint preconditioners.

4.2.2 | Schur complement preconditioners

Schur complement preconditioners are a common choice for saddle point problems. These preconditioners make use of the Schur complement (4.6) to form matrices which use the block structure of the original saddle point matrix and approximate the diagonal or triangular part. Here we consider the block diagonal and block triangular Schur complement preconditioners and approximations to the matrices $\mathbf{L}$ and $\mathbf{H}$ in Table 4.1. Schur complement preconditioners are detailed further in [14, 15, 108].

Block diagonal Schur complement preconditioners

The simplest Schur complement preconditioner is the block diagonal preconditioner

$$\mathcal{P}_D = \begin{bmatrix} \mathbf{D} & 0 & 0 \\ 0 & \mathbf{R} & 0 \\ 0 & 0 & -\tilde{\mathbf{S}} \end{bmatrix}, \tag{4.8}$$

where $\tilde{\mathbf{S}}$ is an approximation to the Schur-complement (4.6) such as those in Table 4.1.

Due to its simple construction and efficacy, the block diagonal Schur complement preconditioner is a popular preconditioner, and used often for saddle point problems arising in fluid dynamics.

Applying this preconditioner to the data assimilation saddle point matrix allows MINRES to be used. However for the purposes of this thesis we apply GMRES to compare the different preconditioners considered. In the full-rank case, GMRES and MINRES are theoretically equivalent in exact arithmetic.

Analysis of the block diagonal Schur complement preconditioners

When applying the block diagonal Schur complement preconditioner to the data assimilation saddle point problem, the resulting preconditioned matrix $\mathcal{A}\mathcal{P}^{-1}$ is of the form

$$\mathcal{A}\mathcal{P}^{-1} = \begin{bmatrix} \mathbf{D} & 0 & \mathbf{L} \\ 0 & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^T & \mathbf{H}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{D}^{-1} & 0 & 0 \\ 0 & \mathbf{R}^{-1} & 0 \\ 0 & 0 & -\tilde{\mathbf{S}}^{-1} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & 0 & -\mathbf{L}\tilde{\mathbf{S}}^{-1} \\ 0 & I_{(N+1)p} & -\mathbf{H}\tilde{\mathbf{S}}^{-1} \\ \mathbf{L}^T\mathbf{D}^{-1} & \mathbf{H}^T\mathbf{R}^{-1} & 0 \end{bmatrix}. \tag{4.9}$$

When taking the exact matrices $\mathbf{S}$, $\mathbf{L}$ and $\mathbf{H}$ for the approximations $\tilde{\mathbf{S}}$, $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{H}}$ respectively, the resulting preconditioned system has three distinct eigenvalues as we observe in Figure 4.2.

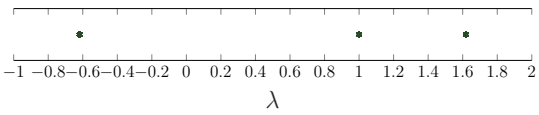


Figure 4.2: Eigenvalues of $\mathcal{A}\mathcal{P}^{-1}$ using the block diagonal Schur complement preconditioner with the exact Schur complement.

Here we see that the eigenvalues of the preconditioned saddle point matrix irrespective of model operator consist of three distinct points, 1, $\frac{1}{2}(1+\sqrt{5})$ and $\frac{1}{2}(1-\sqrt{5})$ as proved in [39, 97]. For this scenario, MINRES (or GMRES) converge in at most three steps. However in general, we must consider approximations to $\mathbf{S}$ which reduces the efficacy of the preconditioner.

Block triangular Schur complement preconditioners

An alternative Schur complement preconditioner is the block triangular Schur complement preconditioner, which unlike the block diagonal one above necessitates the

use of GMRES. The block triangular preconditioner is of the form:

$$\mathcal{P}_T = \begin{bmatrix} \mathbf{D} & 0 & \tilde{\mathbf{L}} \\ 0 & \mathbf{R} & \tilde{\mathbf{H}} \\ 0 & 0 & \tilde{\mathbf{S}} \end{bmatrix}, \tag{4.10}$$

where here as with the block diagonal preconditioner, we consider approximations to $\mathbf{L}$, $\mathbf{H}$, and the Schur complement $\mathbf{S}$.

Triangular preconditioners are among the most effective preconditioners [14], with the first two block rows of $\mathcal{P}$ coinciding with $\mathcal{A}$ when exact $\mathbf{L}$ and $\mathbf{H}$ are chosen.

Analysis of the block triangular Schur complement preconditioners

When applying the block triangular Schur complement preconditioner to the data assimilation saddle point problem, the resulting preconditioned matrix $\mathcal{A}\mathcal{P}^{-1}$ is of the form

$$\begin{aligned} \mathcal{A}\mathcal{P}^{-1} &= \begin{bmatrix} \mathbf{D} & 0 & \mathbf{L} \\ 0 & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^T & \mathbf{H}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{D}^{-1} & 0 & -\mathbf{D}^{-1}\tilde{\mathbf{L}}\tilde{\mathbf{S}}^{-1} \\ 0 & \mathbf{R}^{-1} & -\mathbf{R}^{-1}\tilde{\mathbf{H}}\tilde{\mathbf{S}}^{-1} \\ 0 & 0 & \tilde{\mathbf{S}}^{-1} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{I} & 0 & (\mathbf{L}-\tilde{\mathbf{L}})\tilde{\mathbf{S}}^{-1} \\ 0 & I_{(N+1)p} & (\mathbf{H}-\tilde{\mathbf{H}})\tilde{\mathbf{S}}^{-1} \\ \mathbf{L}^T\mathbf{D}^{-1} & \mathbf{H}^T\mathbf{R}^{-1} & (-\mathbf{L}^T\mathbf{D}^{-1}\tilde{\mathbf{L}} - \mathbf{H}^T\mathbf{R}^{-1}\tilde{\mathbf{H}})\tilde{\mathbf{S}}^{-1} \end{bmatrix}, \end{aligned} \tag{4.11}$$

where we observe that unlike the diagonal Schur complement preconditioner, we retain a term containing $\tilde{\mathbf{L}}$ in $\mathcal{P}^{-1}$, in addition to the $\tilde{\mathbf{L}}^{-1}$ arising from $\tilde{\mathbf{S}}^{-1}$.

When we consider taking exact matrices $\mathbf{S}$, $\mathbf{L}$ and $\mathbf{H}$ for the approximations $\tilde{\mathbf{S}}$, $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{H}}$ respectively, the resulting preconditioned system has one distinct eigenvalue as we observe in Figure 4.3.

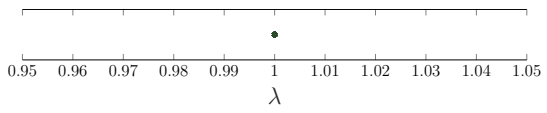


Figure 4.3: Eigenvalues of $\mathcal{A}\mathcal{P}^{-1}$ using the block triangular Schur complement preconditioner with the exact Schur complement.

The spectrum here is precisely $\{1\}$, however the matrix (4.11) (with exact $\tilde{\mathbf{L}}$, $\tilde{\mathbf{H}}$, $\tilde{\mathbf{S}}$) is not diagonalisable [14]. If in (4.10), $\tilde{\mathbf{S}}$ is replaced with $-\tilde{\mathbf{S}}$, the resulting

preconditioned system has two distinct eigenvalues for $\tilde{\mathbf{S}} = \mathbf{S}$, and is diagonalisable.

However this approach is using the exact Schur complement preconditioner, in practice approximations must be taken for $\mathbf{L}$ and $\mathbf{H}$ as presented in Table 4.1.

4.2.3 | Inexact constraint preconditioners

An alternative class of preconditioners are constraint preconditioners. Constraint preconditioners arise from the idea that the preconditioning matrix should have the same block structure as the original saddle point matrix. It is extensively used in the solution of saddle point systems particularly those arising from elliptic PDEs. Here we consider the inexact constraint preconditioner [16, 17, 18], a modification which has an inexact $(1, 2)$ block. In [50, 53] it is noted that this makes for an effective choice of preconditioner to account for the more expensive $(1, 2)$ block in the data assimilation setting:

$$\mathcal{P} = \begin{bmatrix} \mathbf{D} & 0 & \tilde{\mathbf{L}} \\ 0 & \mathbf{R} & \tilde{\mathbf{H}} \\ \tilde{\mathbf{L}}^T & \tilde{\mathbf{H}}^T & 0 \end{bmatrix}, \tag{4.12}$$

provided good approximations are chosen for $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{H}}$. However, this (as with the data assimilation saddle point matrix itself) is an indefinite matrix, and thus using an inexact constraint preconditioner requires the use of GMRES since the resulting preconditioned system is non-symmetric.

Analysis of the inexact constraint preconditioners

Applying the inexact constraint preconditioner to the data assimilation saddle point problem, the eigenvalues of the resulting preconditioned matrix can be bounded using the following result from [17, 18].

Corollary 4.2. *[17, 18] Let* $\mathbf{A}$ *and* $\mathbf{P}$ *be the matrices*

$$\mathbf{A} = \begin{bmatrix} A_1 & B_1^T \\ B_1 & 0 \end{bmatrix}, \quad \textit{and} \quad \mathbf{P} = \begin{bmatrix} A_1 & \tilde{B_1}^T \\ \tilde{B_1} & 0 \end{bmatrix},$$

respectively and assume that $\tilde{B_1}$ *has maximum rank. The eigenvalues* λ *of* $\mathbf{P}^{-1}\mathbf{A}$ *are either one or bounded by*

$$|\lambda - 1| \leq \frac{\|(B_1 - \tilde{B_1})A_1^{-1/2}\|}{\tilde{\sigma_1}},$$

where $\tilde{\sigma}_1$ is the smallest singular value of $\tilde{B}_1 A_1^{-1/2}$, and $\|\cdot\|$ denotes any norm.

Applying this result to the data assimilation saddle point problem, we obtain that the eigenvalues λ of the matrix

$$\begin{bmatrix} \mathbf{D} & 0 & \tilde{\mathbf{L}} \\ 0 & \mathbf{R} & \tilde{\mathbf{H}} \\ \tilde{\mathbf{L}}^T & \tilde{\mathbf{H}}^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{D} & 0 & \mathbf{L} \\ 0 & \mathbf{R} & \mathbf{H} \\ \mathbf{L}^T & \mathbf{H}^T & 0 \end{bmatrix} \tag{4.13}$$

are either one, or bounded by

$$|\lambda - 1| \leq \frac{\left\| \left[(\mathbf{L}^T - \tilde{\mathbf{L}}^T)\mathbf{D}^{-1/2} \quad (\mathbf{H}^T - \tilde{\mathbf{H}}^T)\mathbf{R}^{-1/2} \right] \right\|}{\tilde{\sigma}_1},$$

where $\tilde{\sigma}_1$ is the smallest singular value of $\left[\tilde{\mathbf{L}}^T\mathbf{D}^{-1/2} \quad \tilde{\mathbf{H}}^T\mathbf{R}^{-1/2} \right]$.

In [53], it is shown that when considering the exact approximation $\tilde{\mathbf{L}} = \mathbf{L}$, and taking $\tilde{\mathbf{H}} = 0$, the resulting preconditioned system has eigenvalues

$$\tau = 1 \pm \sqrt{\frac{v^T\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}\mathbf{H}^T v}{v^T\mathbf{R}v}}\ \mathrm{i},$$

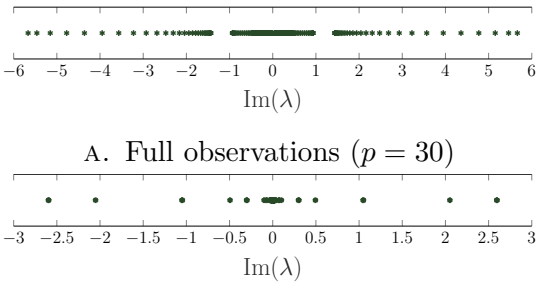
where $v \in \mathbb{R}^{(N+1)p}$. Using the properties of the Rayleigh quotient, we know that the eigenvalues are on a line parallel to the imaginary axis through 1, where the maximum distance from the real axis is given by

$$\sqrt{\frac{\lambda_{\max}(\mathbf{H}\mathbf{L}^{-1}\mathbf{D}\mathbf{L}^{-T}\mathbf{H}^T)}{\lambda_{\min}(\mathbf{R})}}.$$

In Figure 4.4, we plot the imaginary part of the eigenvalues of $\mathcal{A}\mathcal{P}^{-1}$ for the advection-diffusion example with full and partial observations as described in Section 4.2.1.

4.2.4 | Spectral properties of the preconditioned data assimilation saddle point problem

Let us now investigate the spectra of the preconditioned data assimilation saddle point problem for the advection-diffusion example using the approximations for $\mathbf{L}$,$\mathbf{H}$ and $\mathbf{S}$ in Table 4.1. We consider the problem as described in Chapter 3, with the same dimensions as in Figure 4.1, an assimilation window of $30 = N + 1$ timesteps with a state space discretisation using $n = 30$. We consider both full,



A. Full observations ($p = 30$)

B. Partial observations ($p = 3$)

Figure 4.4: Eigenvalues of $\mathcal{A}\mathcal{P}^{-1}$ using the inexact constraint preconditioner with $\tilde{\mathbf{L}} = \mathbf{L}, \tilde{\mathbf{H}} = 0$.

i.e. $p = n$, and partial observations with $p = 3$. For both scenarios we assume the background, observation and model errors have zero mean with covariances $B = 0.01I_n$, $R = 0.01I_p$ and $Q = 10^{-4}I_n$ respectively.

In this section we focus only on the spectra of the preconditioned system using the advection-diffusion example. The shallow water equations and Lorenz examples as presented in Figure 4.1 are qualitatively similar.

Full observations

Prior to preconditioning, the spectra of $\mathcal{A}$ is presented in Figure 4.5. There

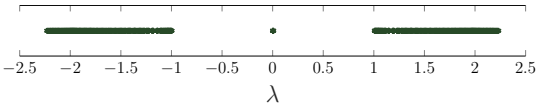
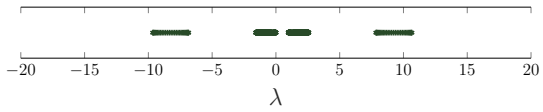


Figure 4.5: Eigenvalues of $\mathcal{A}$ with full observations.

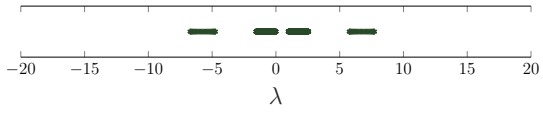
are three distinct groupings of eigenvalues, those near zero, and in the intervals approximately $[-2.25, -1]$ and $[1, 2.25]$ (see Table 4.2).

Block diagonal Schur complement preconditioner

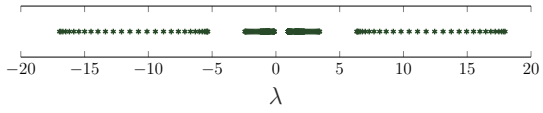
Applying the block diagonal Schur complement preconditioners to this problem with the approximations from Table 4.1, we obtain the spectra in Figure 4.6. We observe that including the exact observation matrix $\mathbf{H}$ results in slightly less spread eigenvalues, but qualitatively the spectra for this preconditioned problem is similar to taking $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = 0$. Taking the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ we observe that the cluster of eigenvalues with larger magnitude are spread over a larger interval than in the previous two preconditioned systems.



A. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = 0$



B. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = \mathbf{H}$

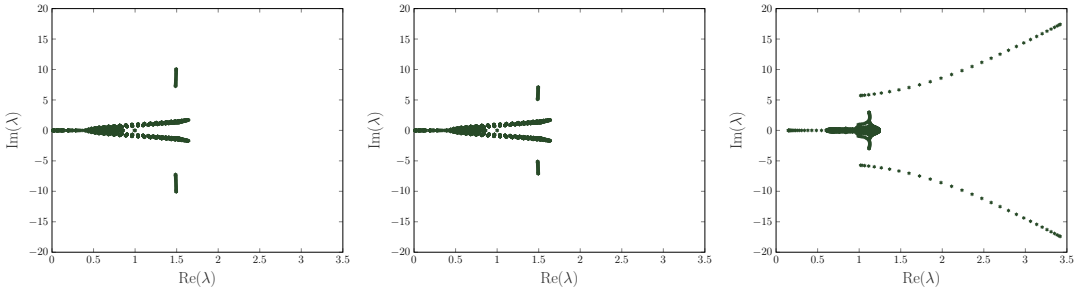


C. $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$

Figure 4.6: Eigenvalues of $\mathcal{A}\mathcal{P}^{-1}$ with full observations using the block diagonal Schur complement preconditioner.

Block triangular Schur complement preconditioner

When using the block triangular Schur complement preconditioner, we observe that the behaviour of the spectra taking $\tilde{\mathbf{L}} = \mathbf{I}$ returns similar results irrespective of choosing $\tilde{\mathbf{H}} = 0$ or $\tilde{\mathbf{H}} = \mathbf{H}$. In contrast, we observe in Figure 4.7 taking the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, the eigenvalues of the preconditioned system are further away from 0 and there are an arc of eigenvalues from approximately $1 \pm -5i$ to 3.5 ± -20. These eigenvalues are more spread out with more distinct clusters.



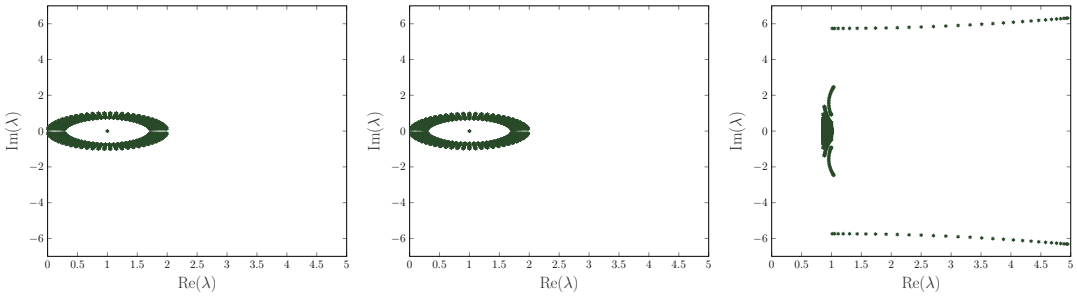
A. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = 0$ B. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = \mathbf{H}$ C. $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$

Figure 4.7: Eigenvalues of $\mathcal{A}\mathcal{P}^{-1}$ with full observations using the block triangular Schur complement preconditioner.

Inexact constraint preconditioner

The eigenvalues of the preconditioned system using the inexact constraint preconditioner are considered in Figure 4.8. We observe that taking the approximation $\tilde{\mathbf{L}} = \mathbf{I}$, the eigenvalues are in a circle of radius 1 centred at $1 + 0i$, with $n = 30$

'spokes' 0.3 long on either side of the real axis and an eigenvalue at 1. In contrast, the eigenvalues for the $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ scenario result in a very different structure. Here we observe a cluster of eigenvalues in arcs with $\mathrm{Re}(\lambda) \approx 1$, in addition to two arcs of eigenvalues from $\pm 6 + i$ to $\pm 6 + 5i$.



A. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = 0$ B. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = \mathbf{H}$ C. $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$

Figure 4.8: Eigenvalues of $\mathcal{A}\mathcal{P}^{-1}$ with full observations using the inexact constraint preconditioner.

Partial observations

We now consider the different resulting spectra if we take partial observations rather than observations of every state, as before, here we consider 10% observations. Prior to preconditioning, the spectra of $\mathcal{A}$ is presented in Figure 4.9. The eigenvalues

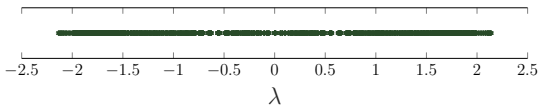


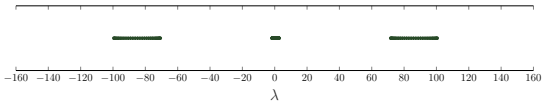
Figure 4.9: Eigenvalues of $\mathcal{A}$ with partial ($p = 3$) observations.

for the problem with partial observations has less clustering than the full observation example. As seen in Table 4.2, the eigenvalues are spread across the whole range $[-2.2, 2.2]$, as a result, we expect the convergence of GMRES to be worse for the observation examples which we consider in Section 4.3.

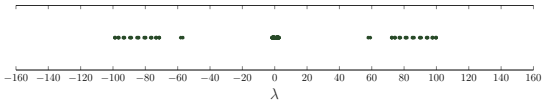
Block diagonal Schur complement preconditioner

When we consider the difference between the spectra of the block diagonal Schur complement preconditioned system for the system with full observation and the partial observations here in Figure 4.10, we observe that the spread of the spectra with partial observations is much greater. Considering the approximation with $\tilde{\mathbf{L}} = \mathbf{I}$, the largest magnitude eigenvalue is approximately 100 in contrast to 10

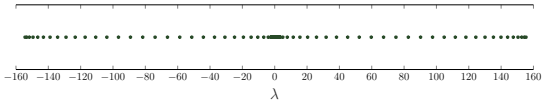
in Figure 4.6. We observe some clustering with the addition of the observation operator $\mathbf{H}$, however the spectra are qualitatively similar. Taking $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ the spectra is spread over a larger interval $[-160, 160]$, though with a relatively small number of distinct eigenvalues.



A. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = 0$



B. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = \mathbf{H}$

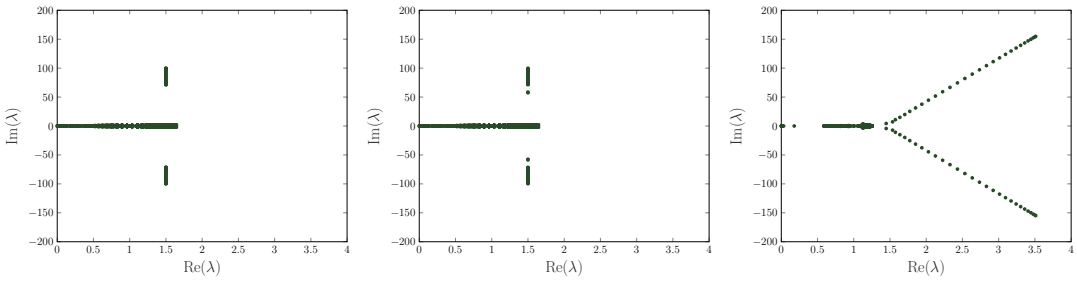


C. $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$

Figure 4.10: Eigenvalues of $\mathcal{A}\mathcal{P}^{-1}$ with partial observations using the block diagonal Schur complement preconditioner.

Block triangular Schur complement preconditioner

For the block triangular Schur complement preconditioner, the spectra of the preconditioned system is similar for the full and partial observations for the eigenvalues with $\mathrm{Im}(\lambda) < 2$. However for those with $\mathrm{Im}(\lambda) > 2$ in the full observations example, the corresponding eigenvalues are significantly larger in Figure 4.11.

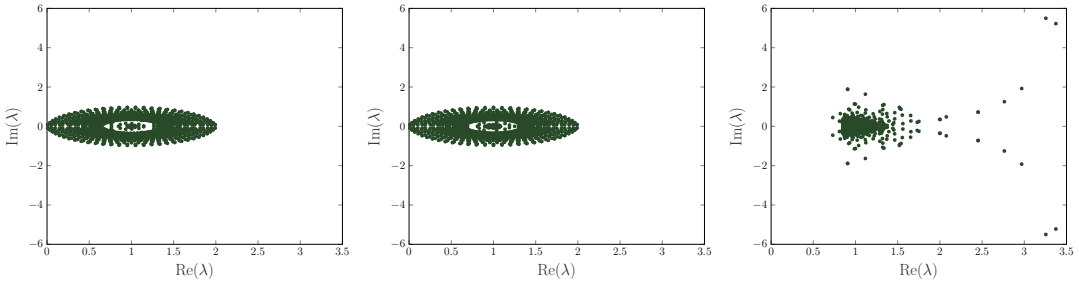


A. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = 0$ B. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = \mathbf{H}$ C. $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$

Figure 4.11: Eigenvalues of $\mathcal{A}\mathcal{P}^{-1}$ with partial observations using the block triangular Schur complement preconditioner.

Inexact constraint preconditioner

As with the full observation example, the spectra of the preconditioned systems using the inexact constraint preconditioner with the approximation $\tilde{\mathbf{L}} = \mathbf{I}$ are in a circle of radius 1 centred at $1 + 0i$, although here in Figure 4.12 there is less clustering, with a greater number of distinct eigenvalues and less structure than in Figure 4.8. Taking $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, the behaviour of the spectra is very different to the full observations example, and we see many distinct eigenvalues with majority of these clustered around $1 + i$.



A. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = 0$ B. $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{H}} = \mathbf{H}$ C. $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$

Figure 4.12: Eigenvalues of $\mathcal{A}\mathcal{P}^{-1}$ with partial observations using the inexact constraint preconditioner.

Summary

From the spectra of the preconditioned systems we have observed in this section, we see that considering the data assimilation saddle point problem with partial observations results in significantly less clustering for the eigenvalues of the preconditioned systems. This results in a harder problem, and we expect to observe the examples with partial observations in the following section needing more iterations to reach convergence than when taking full observations.

For each of the three types of preconditioner considered in this section, we have observed that applying the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ results in greater clustering of the eigenvalues of the preconditioned systems. The structure of the spectra is quite different to that obtained when considering the approximation $\tilde{\mathbf{L}} = \mathbf{I}$. Similar spectra for taking $\tilde{\mathbf{H}} = 0$ and $\tilde{\mathbf{H}} = \mathbf{H}$ is observed, with slightly tighter clustering when including $\tilde{\mathbf{H}} = \mathbf{H}$. As such we would expect to see a slight improvement in the efficacy of the preconditioner when using $\tilde{\mathbf{H}} = \mathbf{H}$ in the following numerical examples.

4.3 | Numerical results

Let us now compare these preconditioners using the approximations from Table 4.1 for $\mathbf{L}$ and $\mathbf{H}$. In this section we consider the three examples from Chapter 3 and in the preceding sections: the 1D advection-diffusion example, the linearised 2D shallow water equations, and the Lorenz-95 problem.

For these examples, as above, we consider an assimilation window of $30 = N + 1$ timesteps for all three problems, taking a state space discretisation using $n = 30$ for the advection-diffusion and Lorenz-95 problems, and $n = 27$ for the shallow water equations example. Furthermore, we assume the background, observation and model errors have zero mean with covariances $B = 0.01I_n$, $R = 0.01I_p$ and $Q = 10^{-4}I_n$ respectively. We consider both full observations, i.e. $p = n$, and partial observations with $p = 3$.

As we saw in Section 4.2, when using the exact Schur complement in the Schur complement preconditioners, we would expect GMRES to converge in three or less iterations [14, 97]. However we are using the approximations presented in Table 4.1 and as such will not achieve such fast convergence.

In the following sections we present the norm of the residual computed in GMRES at each iteration for the different choices of preconditioner.

4.3.1 | Advection-diffusion

As our first example we present the advection-diffusion problem as introduced in Section 3.3.1. We take a state space discretisation of $n = 30$ with $N + 1 = 30$ timesteps which results in a saddle point system of size $(1800 + 30p) \times (1800 + 30p)$.

Full observations

In Figure 4.13, we compare the convergence of GMRES for the different preconditioners using the advection-diffusion example with full observations which results in a 2700×2700 matrix.

We see that for the first 50 iterations, using no preconditioner leads to the best convergence, at which point the inexact constraint preconditioner with $\tilde{\mathbf{L}} = \hat{\mathbf{L}}, \tilde{\mathbf{H}} = 0$ achieves a lower residual.

Here we see that for all three types of preconditioner the best results are obtained for $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{H}}$ are taking $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$. Indeed, taking $\tilde{\mathbf{L}} = \mathbf{I}$ we see similar plots for the convergence with or without the inclusion of $\mathbf{H}$. This is not unexpected,

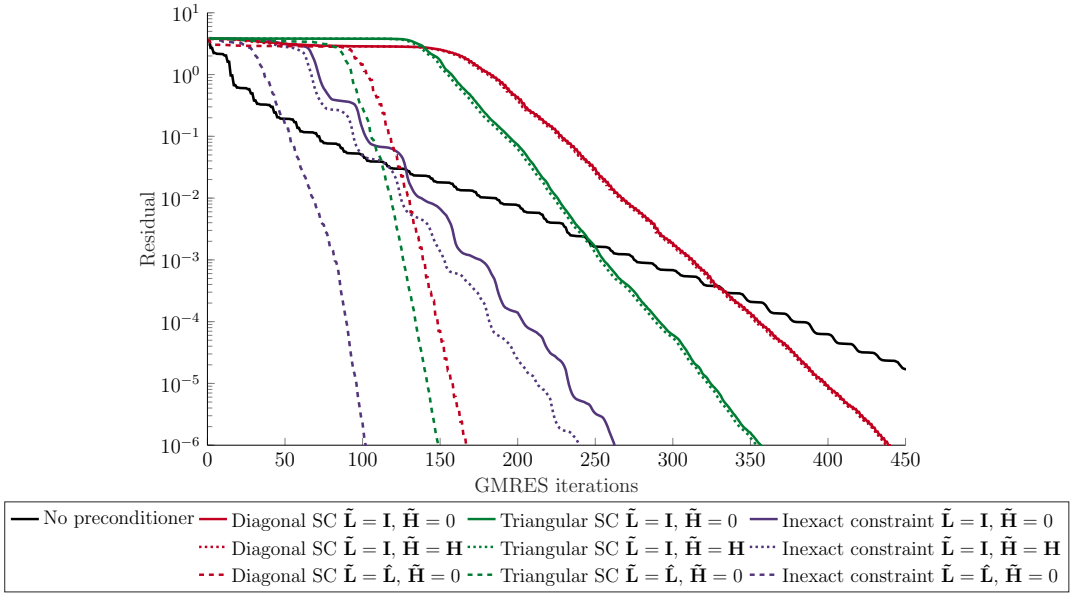


Figure 4.13: GMRES residual with different preconditioners for the 2700×2700 advection-diffusion example with full observations.

as we saw in Section 4.2.4, the spectra for the preconditioned problem when using $\tilde{\mathbf{L}} = \mathbf{I}$ was very similar irrespective of the choice of $\tilde{\mathbf{H}}$. For the inexact constraint preconditioner, taking $\tilde{\mathbf{H}} = \mathbf{H}$ does result in a slight improvement over using $\tilde{\mathbf{H}} = 0$ however it is not as significant as considering $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ as some model information is included using this approximation.

The two Schur complement preconditioners with $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ are slightly less effective than the inexact constraint preconditioners with $\tilde{\mathbf{L}} = \mathbf{I}$, and all result in higher residuals than not using a preconditioner for the first 120 iterations, however after this number of iterations, the Schur complement preconditioners converge faster.

Partial observations

Let us now consider partial observations. Here we keep the rest of the example as before, but take 10% ($p = 3$) observations. The corresponding observation error covariance matrix we take to be $R = 0.01I_p$ as before.

We observe that the convergence for this problem with no preconditioner is significantly slower than the previous example. Taking partial observations results in a harder problem in Figure 4.14 than Figure 4.13. We observed in Section 4.2.4 that there is less clustering of the eigenvalues for the partial observation case which typically results in slower convergence of methods such as GMRES. Whilst in the previous example, the residual for the unpreconditioned problem was 10^{-4} after 400 iterations, here it takes approximately 1200 to reach the same level, despite being a

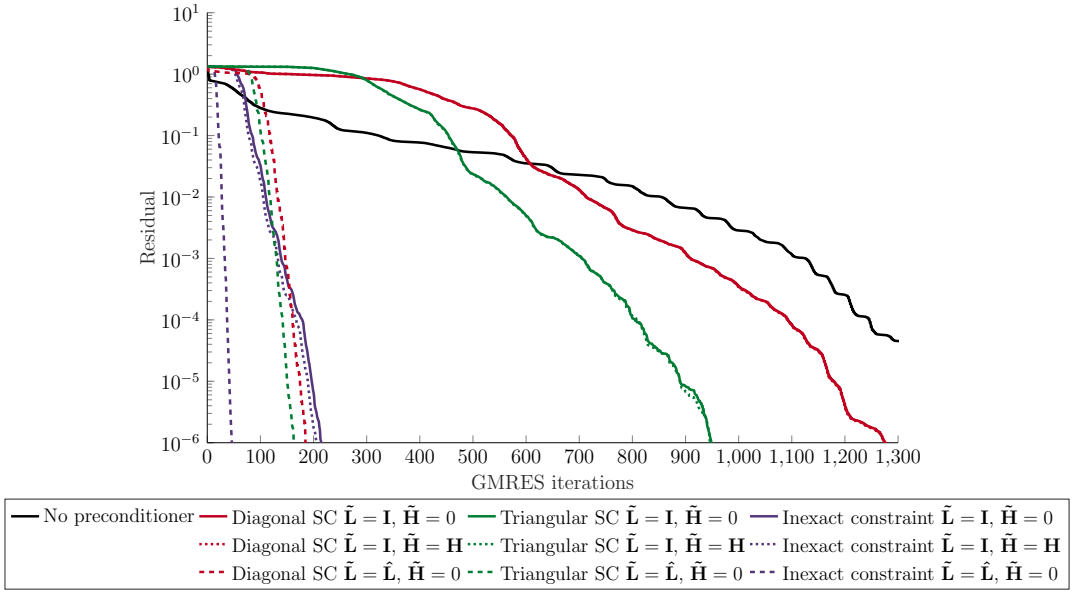


Figure 4.14: GMRES residual with different preconditioners for the 1890×1890 advection-diffusion example with partial observations.

smaller problem size. This is not too surprising given the eigenstructure observed in Section 4.2.4, where the eigenvalues of the system were spread over a larger interval, and closer to 0.

The Schur complement preconditioners with $\tilde{\mathbf{L}} = \mathbf{I}$ are ineffective in comparison to the other preconditioners presented here, taking over 900 iterations for the residual to be smaller than 10^{-6}, and over 500 iterations to be more effective than not using a preconditioner.

As with the previous example, the most effective preconditioner is the inexact constraint preconditioner taking the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, with the residual reaching 10^{-6} after only 50 iterations. This is more effective than the example with full observations. The other approximations for the inexact constraint preconditioner are also effective for the first 100 iterations, at which point the two Schur complement preconditioners with $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ result in slightly lower residuals.

4.3.2 | Shallow water equations

We now consider the two dimensional shallow water equations example from Section 3.3.2. To obtain a similarly sized example as above, we take a state space discretisation of $n = 27$ with $N + 1 = 30$ timesteps resulting in a saddle point system of size $(1620 + 30p) \times (1620 + 30p)$.

Full observations

As before we first consider full ($p = 27$) observations, resulting in a saddle point matrix of size 2430×2430.

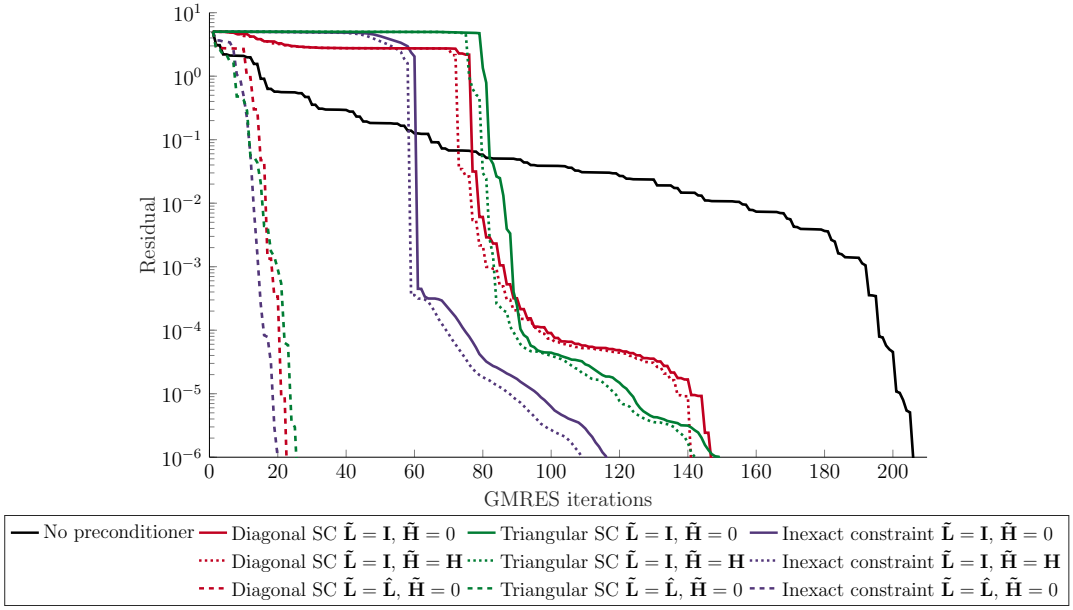


Figure 4.15: GMRES residual with different preconditioners for the 2430×2430 shallow water equations example with full observations.

In the shallow water equations example in Figure 4.15 we see that the convergence of GMRES is significantly faster than the advection-diffusion example with full observations. Here the residual for the unpreconditioned problem reaches 10^{-6} after only 200 iterations. As with the advection-diffusion examples we see that taking the approximation $\tilde{\mathbf{L}} = \mathbf{I}$ results in similar convergence rates for the preconditioners, irrespective of the inclusion of $\tilde{\mathbf{H}} = \mathbf{H}$ in contrast to 0. The residuals for these six preconditioners stagnate with little change for the first 50 iterations before a significant drop in the residual.

The three preconditioners where we take the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ result in very similar convergence to one another, with the inexact constraint preconditioner marginally superior, all three achieving a residual smaller than 10^{-6} after only 20 iterations. Despite this, for the first 5 iterations there is no improvement in the residual over using no preconditioner.

Partial observations

Let us now consider partial observations for the SWE example taking $p = 3$, with the rest of the example as before. This gives a saddle point matrix of size

1701×1701.

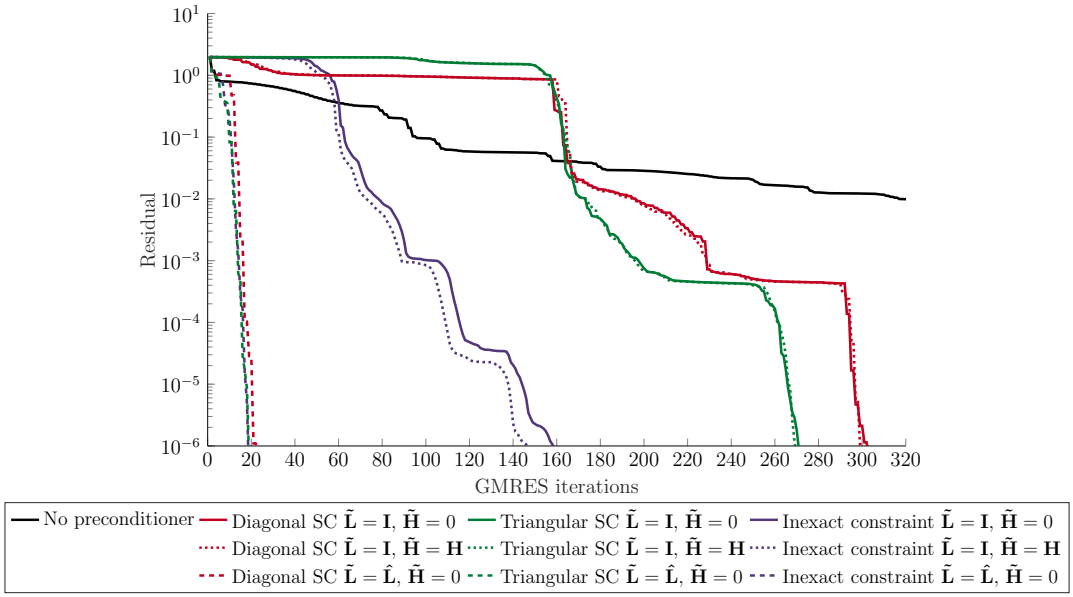


Figure 4.16: GMRES residual with different preconditioners for the 1701×1701 shallow water equations example with partial observations.

As with the advection-diffusion example, here taking partial observations results in a harder problem, and thus more iterations needed as the eigenvalues are less clustered than the problem in Figure 4.16. We observe that as in the previous example, all three preconditioners taking the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ are equally the most effective, taking only 20 iterations to achieve a residual of 10^{-6}. The remaining inexact constraint preconditioners are marginally less effective than in the full observations example, however have a more gradual reduction in the residual. In contrast the two Schur complement preconditioners with the approximation $\tilde{\mathbf{L}} = \mathbf{I}$ are significantly less effective in the partial observations setting of Figure 4.16, with both less effective than no preconditioner for the first 160 iterations.

The efficacy of the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ for the shallow water equations example is likely due to the closeness of the eigenvalues of the model matrix M to 1 as seen in Figure 3.5 b). The inexact constraint preconditioner with this approximation achieved the best results for the shallow water equations examples.

4.3.3 | Lorenz system

For our final example, let us consider the nonlinear Lorenz-95 system example we considered in Section 3.4.2. As with the advection-diffusion example, we take $n = 30$ states, and $N + 1 = 30$ assimilation timesteps, with the covariance matrices as in

our previous examples, resulting in a saddle point system of size $(1800 + 30p) \times (1800 + 30p)$.

This is a nonlinear example and as such requires multiple inner loops during incremental 4D-Var. We consider the first linearisation, and use this to illustrate the behaviour of applying preconditioned GMRES to the linear system for the following examples.

Full observations

Taking full ($p = 30$) observations at each timestep, the size of the saddle point matrix is 2700×2700 and the residuals for GMRES with the different preconditioners and approximations are given in Figure 4.17.

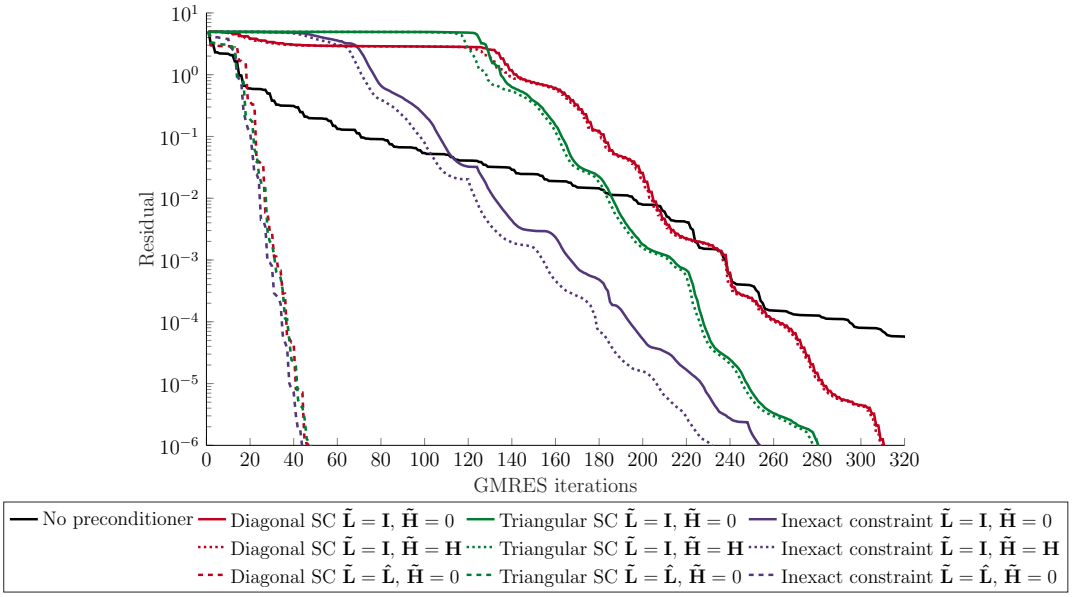


Figure 4.17: GMRES residual with different preconditioners for the 2700×2700 Lorenz-95 example with partial observations.

In this example, as with the shallow water equations example, the three preconditioners with the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ result in similar convergence plots, and are significantly better than the alternatives presented here. Although once again, for the first 20 iterations, using no preconditioner results in the smallest residual. The remaining preconditioners taking the approximation $\tilde{\mathbf{L}} = \mathbf{I}$ are less effective than not using a preconditioner to begin with, but ultimately do obtain a lower residual.

Partial observations

As our final example, we consider the Lorenz system with partial ($p = 3$) observations resulting in a saddle point system of size 1890×1890. This is a harder problem than the full observations example above despite the smaller size, due to the reduced clustering of the eigenvalues when considering partial observations as seen in Section 4.2.4.

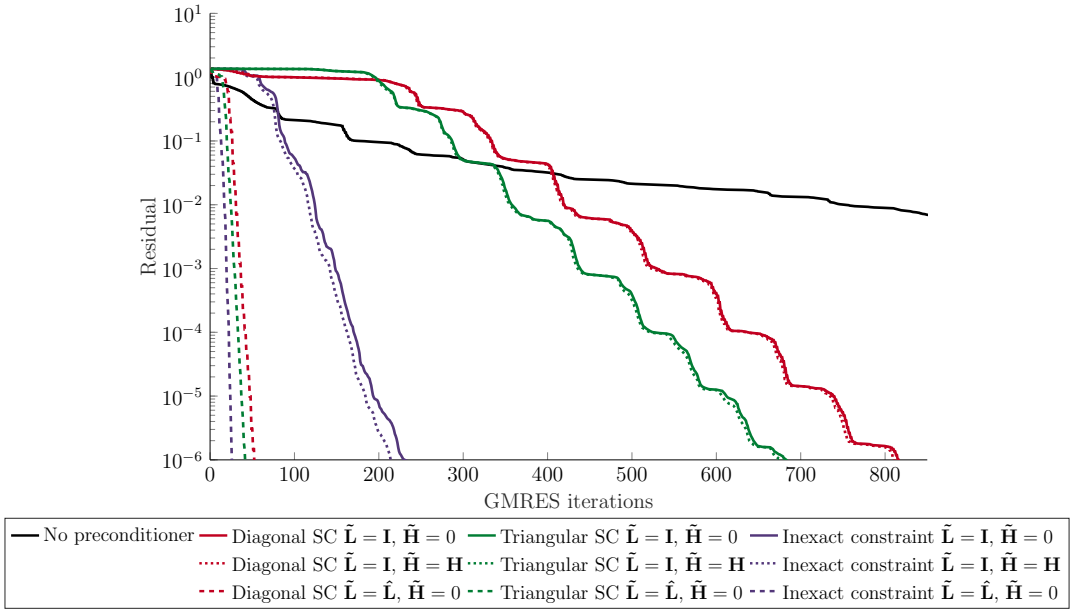


Figure 4.18: GMRES residual with different preconditioners for the 1890×1890 Lorenz-95 example with partial observations.

Here we observe that the most effective preconditioner in Figure 4.18, as with all our previous examples, is the inexact constraint preconditioner with the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$. The block triangular, and block diagonal Schur complement preconditioners with the same approximations are similarly effective, and take less than 100 iterations to reach a residual of 10^{-6}. The remaining inexact constraint preconditioners are slightly more effective for the partial observations Lorenz example here than the full observations above, however is initially worse than not using a preconditioner.

The four Schur complement preconditioners with $\tilde{\mathbf{L}} = \mathbf{I}$ are all significantly less effective than in the full observations example, and have a higher residual than not applying a preconditioner for the first 300 timesteps

4.3.4 | Summary

From these examples we see that across all different examples considered, the most effective preconditioner was the inexact constraint preconditioner with the approximations $\tilde{\mathbf{L}} = \hat{\mathbf{L}} = (I_{N+1} + C) \otimes I_n$ and $\tilde{\mathbf{H}} = 0$. The block triangular and block diagonal Schur complement preconditioners with these approximations were the next most effective, although in the advection-diffusion example, there was a larger difference between these and the inexact constraint preconditioner.

The inclusion of the true observation operator, taking $\tilde{\mathbf{H}} = \mathbf{H}$ did not appear to make a large difference on the efficacy of the preconditioners when using the approximation $\tilde{\mathbf{L}} = \mathbf{I}$, particularly when applying the Schur complement preconditioners. It was the Schur complement preconditioners with the approximation $\tilde{\mathbf{L}} = \mathbf{I}$ which were the least effective. The two inexact constraint preconditioners with $\tilde{\mathbf{L}} = \mathbf{I}$ were more effective compared to the other approaches when considering the partial observations. In these examples, the inexact constraint preconditioners with $\tilde{\mathbf{L}} = \mathbf{I}$ were closer to the performance of the Schur complement preconditioners with $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, during the initial iterations.

When considering partial observations, we see that the number of iterations needed for the residual to be 10^{-6} is significantly larger when not applying a preconditioner, which matches with the greater spread in eigenvalues observed in Figure 4.1. This reduction in efficacy is also apparent for the Schur complement preconditioners with the approximation $\tilde{\mathbf{L}} = \mathbf{I}$. The remaining preconditioners were less affected by the reduction in observations, and we observed similar plots for the residuals. Generally, as observed in Figures 4.13-4.18, the inexact constraint preconditioner with the approximations $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$ worked most effectively.

In the next sections we shall see how the efficacy of the preconditioners changes when solving the data assimilation saddle point problem using the low-rank GMRES from Chapter 3 rather than GMRES as used here.

4.4 | Preconditioning the data assimilation saddle point problem for low-rank GMRES

The low-rank GMRES method introduced in Chapter 3 brings new requirements to consider when implementing preconditioners.

In order to maintain the low-rank structure we wish to write the preconditioned

problem in Kronecker form, however we must also consider the inverse of the preconditioner which must be written in Kronecker form as well. It is the implementation of the inverse in Kronecker form which allows us to write this as a simple matrix multiplication as in (3.41) for the saddle point matrix.

We recall from Chapter 3, the matrices within the saddle point matrix can be written in Kronecker form as follows:

$$\begin{aligned}\mathbf{D} &= E_1 \otimes B + E_2 \otimes Q,\\ \mathbf{R} &= I \otimes R,\\ \mathbf{H} &= I \otimes H,\\ \mathbf{L} &= I \otimes I + C \otimes M.\end{aligned}$$

The approximations introduced in the first half of this chapter can be written in Kronecker form, and in Table 4.3 we present the Kronecker forms for the approximations to $\mathbf{L}$, $\mathbf{H}$ and $\mathbf{S}^{-1}$ introduced in Table 4.1. Here $\hat{\mathbf{L}} = (I_{N+1}+C)\otimes I_n$ as previously. Furthermore we define the matrices $F_B = HBH^T+R$ and $F_Q = HQH^T+R$.

$\tilde{\mathbf{L}}$	$\tilde{\mathbf{H}}$	$\tilde{\mathbf{S}}^{-1}$
$\mathbf{I}$	0	$-E_1 \otimes B - E_2 \otimes Q$
$\mathbf{I}$	$\mathbf{H}$	$E_1 \otimes (-B + BH^T F_B^{-1} HB) + E_2 \otimes (-Q + QH^T F_Q^{-1} HQ)$
$\hat{\mathbf{L}}$	0	$-(I+C)^{-1}E_1(I+C)^{-T} \otimes B - (I+C)^{-1}E_2(I+C)^{-T} \otimes Q$

Table 4.3: Table of approximations for $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{H}}$ and the resulting Schur complement inverse in Kronecker form.

If we consider a more general approximation $\tilde{\mathbf{L}}$ to $\mathbf{L}$, of the form $\tilde{\mathbf{L}} = I_{N+1} \otimes I_n + C \otimes \tilde{M}$, the resulting inverse in Kronecker form contains a large number of terms, and hence we must consider truncation to tractably apply the inverse. In Section 4.6 we consider the exact $\mathbf{L}$ and approximate the inverse through truncation.

As noted above, we wish to implement the inverse of the preconditioner in Kronecker form in order to apply the preconditioner through simple matrix multiplication as in (3.41). This is implemented within LR-GMRES (Algorithm 1) as the `Aprec` function.

To illustrate a possible choice of the this `Aprec` function, we consider the block

diagonal Schur complement preconditioner with $\tilde{\mathbf{L}} = \mathbf{I}, \tilde{\mathbf{H}} = 0$

$$\begin{aligned}\mathcal{P}^{-1} &= \begin{bmatrix} \mathbf{D}^{-1} & 0 & 0 \\ 0 & \mathbf{R}^{-1} & 0 \\ 0 & 0 & -\tilde{\mathbf{S}}^{-1} \end{bmatrix} \\ &= \begin{bmatrix} E_1 \otimes B^{-1} + E_2 \otimes Q^{-1} & 0 & 0 \\ 0 & I_{N+1} \otimes R^{-1} & 0 \\ 0 & 0 & E_1 \otimes B + E_2 \otimes Q \end{bmatrix}.\end{aligned}$$

The application of this $\mathcal{P}^{-1}$ using simple matrix multiplication is shown in Algorithm 4.

Algorithm 4: Block diagonal Schur complement preconditioner $\tilde{\mathbf{L}} = \mathbf{I}$, $\tilde{\mathbf{H}} = 0$ (`Aprec`)

Input: $W_{11}, W_{12}, W_{21}, W_{22}, W_{31}, W_{32}$
Output: $Z_{11}, Z_{12}, Z_{21}, Z_{22}, Z_{31}, Z_{32}$
$Z_{11} = [B^{-1}W_{11}, \quad Q^{-1}W_{11}]$,
$Z_{12} = [E_1 W_{12}, \quad E_2 W_{12}]$,
$Z_{21} = R^{-1}W_{21}$,
$Z_{22} = W_{22}$,
$Z_{31} = [BW_{31}, \quad QW_{31}]$,
$Z_{32} = [E_1 W_{32}, \quad E_2 W_{32}]$

An alternative method for implementing the Schur complement approximation $\tilde{\mathbf{S}} = -\tilde{\mathbf{L}}\mathbf{D}^{-1}\tilde{\mathbf{L}}$, with a $\tilde{\mathbf{L}}$ of the form $(I \otimes I + C \otimes \tilde{M})$, whilst retaining a low-rank form is detailed in [125]. There the relationship between the Kronecker product and Sylvester equations is exploited. In order to solve $-\tilde{\mathbf{S}} Z_{31} Z_{32}^T = W_{31} W_{32}^T$, the Kronecker form

$$(I \otimes I + C^T \otimes \tilde{M}^T)(E_1 \otimes B^{-1} + E_2 \otimes Q^{-1})(I \otimes I + C \otimes \tilde{M}) \mathrm{vec}\left(Z_{31} Z_{32}^T\right) = \mathrm{vec}\left(W_{31} W_{32}^T\right),$$

is written as two consecutive Sylvester equations. These resulting Sylvester equations are solved one after the other using a low-rank solver such as an ADI [10, 13] or Krylov [117] method to generate a low-rank approximation $X_{31} X_{32}^T$. However we do not employ this approach here.

4.5 | Low-rank numerical results

Let us now compare the preconditioners introduced in Section 4.2 using LR-GMRES. In this section we consider the same three examples from Section 4.3: the 1D

advection-diffusion example, the linearised 2D shallow water equations, and the Lorenz-95 problem.

For these examples, we consider an assimilation window of $30 = N + 1$ timesteps for all three problems, taking a state space discretisation with $n = 30$ for the advection-diffusion and Lorenz-95 problems, and $n = 27$ for the shallow water equations example. The background, observation and model errors are assumed to have zero mean with covariances $B = 0.01I_n$, $R = 0.01I_p$ and $Q = 10^{-4}I_n$ respectively. As before we consider both full observations, i.e. $p = n$, and partial observations with $p = 3$.

In the following sections we present the norm of the residual computed in LR-GMRES at each iteration for the different choices of preconditioner. As in Section 3.3, we consider different ranks for the method, taking $r = 20$ and $r = 5$.

4.5.1 | Advection-diffusion

For our first example we present the advection-diffusion problem as introduced in Section 3.3.1. We take a state space discretisation of $n = 30$ with $N + 1 = 30$ timesteps which results in a saddle point system of size $(1800 + 30p) \times (1800 + 30p)$, and an update vector δx with 900 entries, the low-rank solutions will have $60r$ entries in total. We note that due to the small illustrative problem size, the solution itself has a larger number of matrix entries for $r = 20$, but is still a lower rank.

Full observations

Let us first consider taking full observations for this example. In Figure 4.19 we present the LR-GMRES residuals for $r = 20$ and $r = 5$.

The first observation we make from these figures is that majority of methods appear to stagnate in terms of lowering the LR-GMRES residual after a number of iterations. This is likely due to the truncation within the LR-GMRES algorithm. During LR-GMRES, the truncation process selects only the most important modes, e.g. the ones belonging to larger eigenvalues, ignoring the smaller ones. Therefore, the low-rank approach itself acts like a regularisation, and hence in some sense like a projected preconditioner.

For both choices of r, the most effective preconditioners are the inexact constraint preconditioners, however the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ works better in the rank 5 example, whilst taking $\tilde{\mathbf{L}} = \mathbf{I}$ is more effective in the larger $r = 20$ case. The remaining preconditioners do not see significant improvement in the level of the residual, with only small improvements over the 500 iterations consider in Figure 4.19. This is

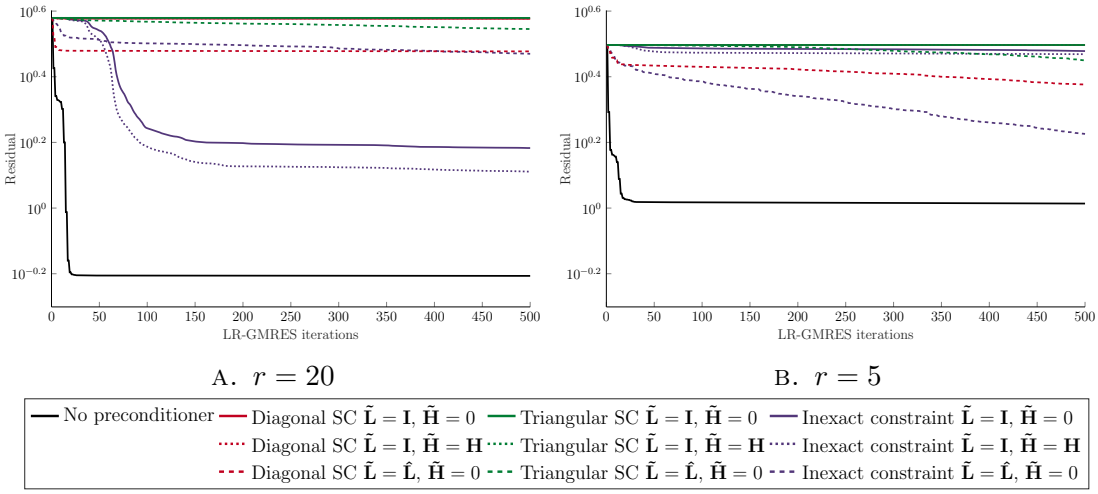


Figure 4.19: LR-GMRES residual with different preconditioners for the 2700×2700 advection-diffusion example with full observations ($r = 20$, $r = 5$).

likely due to the stagnation mentioned above. For this problem and the preconditioners we considered here, the most effective choice of preconditioner is not applying one, with the low-rank approach itself acting like a projected preconditioner.

Partial observations

We now consider partial observations for the advection-diffusion example, taking $p = 3$ observations at each timestep.

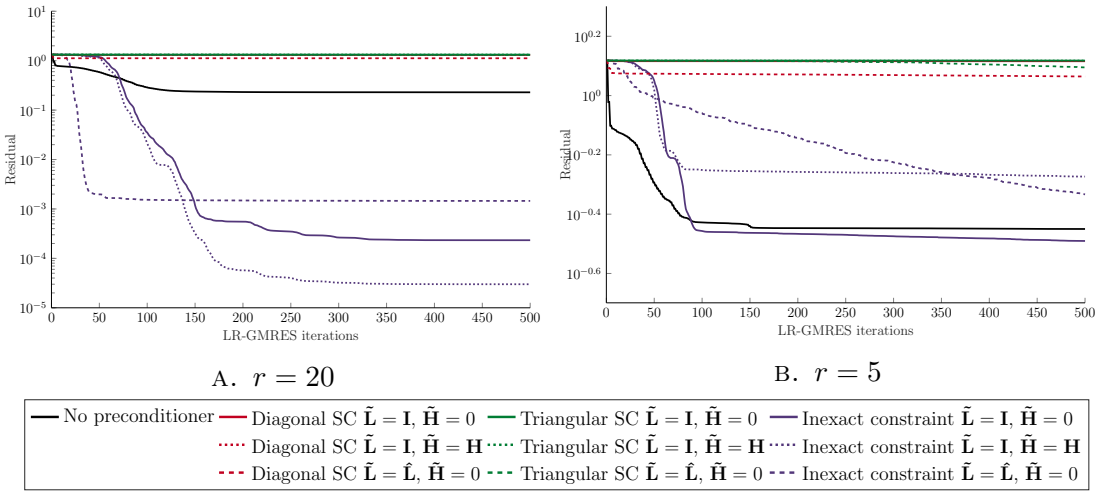


Figure 4.20: LR-GMRES residual with different preconditioners for the 1890×1890 advection-diffusion example with partial observations ($r = 20$, $r = 5$).

In Figure 4.20 we see that the efficacy of preconditioners is significantly better

taking $r = 20$ over $r = 5$. For both choices of rank we see that the Schur complement preconditioners are not effective and result only in a minor improvement of the residual. As in Figure 4.19 we observe stagnation of the residuals, for some preconditioning approaches this occurs sooner than others. The inexact constraint preconditioner with the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ is initially the most effective choice of preconditioner for the $r = 20$ example, however stagnates after 50 iterations, whilst taking $r = 5$ the reduction in the residual is more gradual using this preconditioner. The two inexact constraint preconditioners with $\tilde{\mathbf{L}} = \mathbf{I}$ exhibit similar behaviour to one another for both examples, however the inclusion of $\tilde{\mathbf{H}} = \mathbf{H}$ causes the stagnation of the approach to occur at a different level of residual.

As with the GMRES example in Figure 4.14, we see that initially no preconditioner is most effective for the first 20 iterations, at which point the inexact constraint preconditioner taking the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ has the best convergence when considering $r = 20$. For the $r = 5$ example, the low-rank method acting like a projected preconditioner means we observe that using no preconditioner is most effective choice.

4.5.2 | Shallow water equations

We now consider the two dimensional shallow water equations example from Section 3.3.2. Taking the same dimensions as in Section 4.3.2, we have a state space discretisation of $n = 27$ with $N + 1 = 30$ timesteps.

Full observations

As before we first consider full ($p = 27$) observations, with two choices of rank $r = 20$ and $r = 5$.

Here we observe similar behaviour to the advection-diffusion example with full observations, with majority of the preconditioners being less effective than considering the unpreconditioned saddle point system. The exception here is the inexact constraint preconditioner with the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ for the $r = 20$ example in Figure 4.21 a), where the preconditioned system achieves a lower residual after 40 iterations of LR-GMRES. The remaining preconditioners which see a slight improvement stagnate at a similar residual to one another for both choices of r.

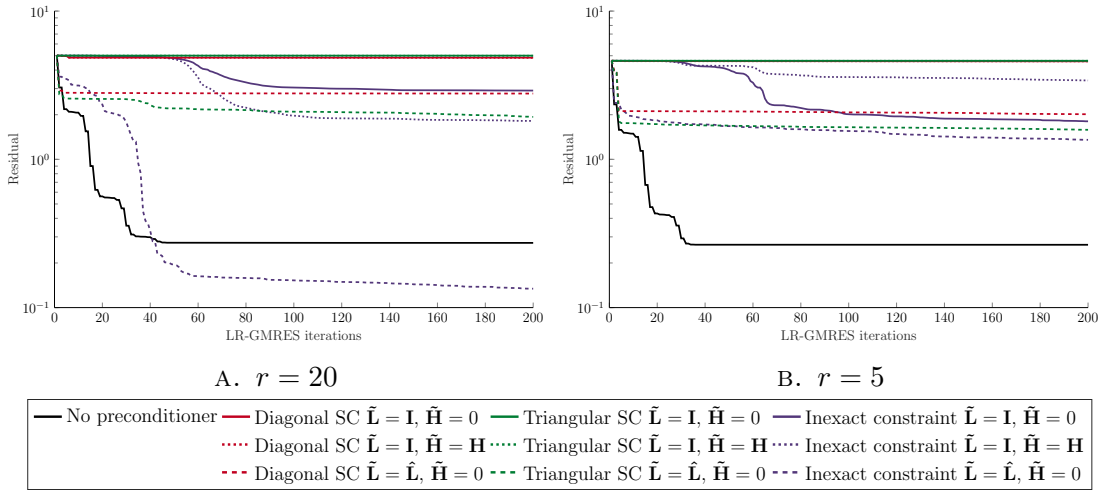


Figure 4.21: LR-GMRES residual with different preconditioners for the 2430×2430 shallow water equations example with full observations ($r = 20$, $r = 5$).

Partial observations

In Figure 4.22 we consider partial observations for the shallow water equations example, taking $p = 3$ observations at each timestep.

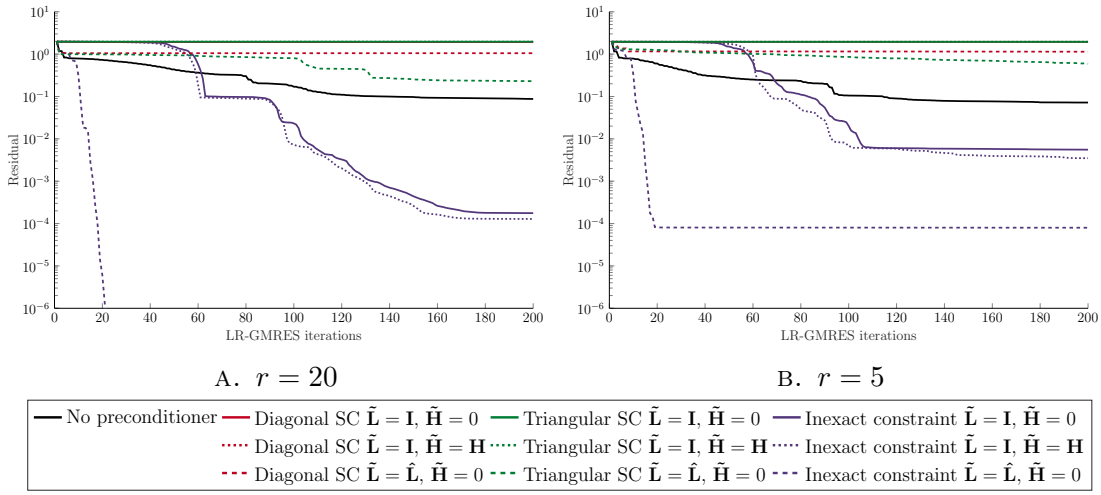


Figure 4.22: LR-GMRES residual with different preconditioners for the 1701×1701 shallow water equations example with partial observations ($r = 20$, $r = 5$).

Here we observe the most effective results for preconditioning the LR-GMRES method. We see that applying the inexact constraint preconditioner with the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$ converges to a level of 10^{-6} in 20 iterations for the $r = 20$ example, and 10^{-4} for the $r = 5$ case (at which point it stagnates). The other two inexact constraint preconditioners are also effective, though taking a

greater number of iterations, and not reaching a residual as small before stagnation. In Figure 4.22 a) we also witness an improvement of the block triangular Schur complement preconditioner taking $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, however it is still less effective than not applying a preconditioner. Initially no preconditioner results in the best convergence for the first 10 iterations, at which point the inexact constraint preconditioner achieves a lower residual.

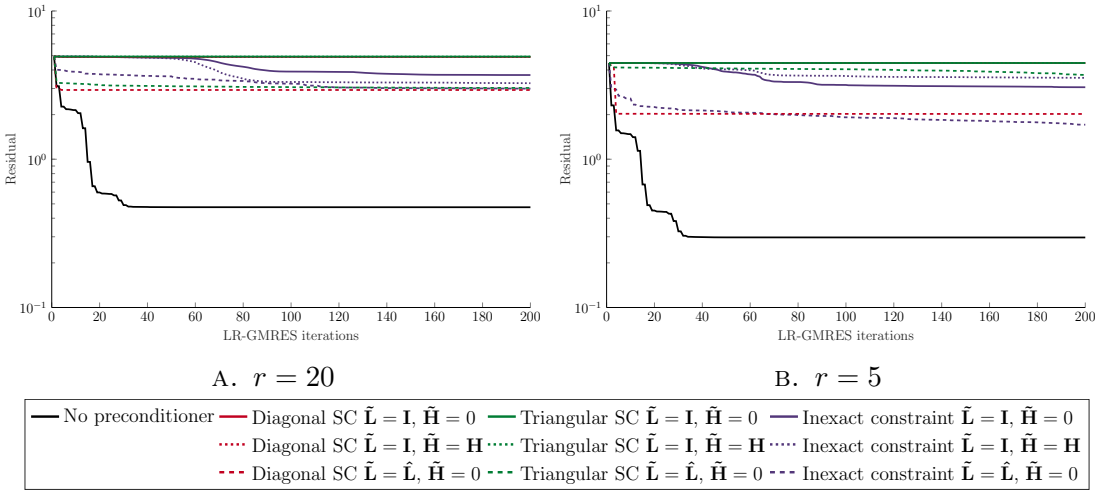
4.5.3 | Lorenz system

For our final example, we again consider the nonlinear Lorenz-95 system introduced in Section 3.4.2. As before, we take $n = 30$ states, and $N + 1 = 30$ assimilation timesteps.

For this example we consider the first linearisation used in incremental 4D-Var as in Section 4.3.3, and use this to investigate applying preconditioners within LR-GMRES to the linear system for the following examples.

Full observations

Taking full ($p = 30$) observations at each timestep, we consider the residuals obtained by applying the different preconditioners and approximations $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{H}}$ to the saddle point problem solved using LR-GMRES in Figure 4.23.



a. $r = 20$ b. $r = 5$

Figure 4.23: LR-GMRES residual with different preconditioners for the 2700×2700 Lorenz-95 example with full observations ($r = 20$, $r = 5$).

Here, as with the full observations examples for the advection-diffusion and shallow water equations problems, we see that the preconditioning approaches all stag-

nate quite early, with no method resulting in significant improvements to the residual. The preconditioners using the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ provide the lowest residuals of the preconditioned problems, however not applying a preconditioner is as with the advection-diffusion example in Figure 4.19 the most effective choice.

Partial observations

Let us now consider partial ($p = 3$) observations for the Lorenz problem. We present the residuals for LR-GMRES applying different preconditioners in Figure 4.24.

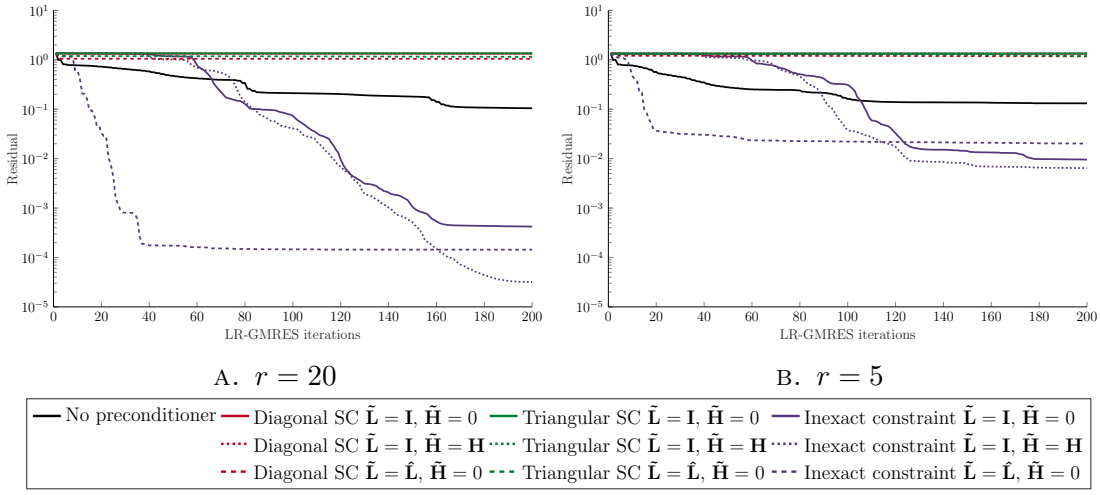


Figure 4.24: LR-GMRES residual with different preconditioners for the 1890×1890 Lorenz-95 example with partial observations ($r = 20$, $r = 5$).

The residuals in this example display similar behaviour to the other examples we have considered. The Schur complement preconditioners are not effective for this problem, stagnating and not improving significantly from the first few iterations. Initially the most effective preconditioner is the inexact constraint preconditioner with the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ for both the $r = 5$ and $r = 20$ examples, however for the smaller choice of rank, the preconditioner stagnates after fewer iterations of LR-GMRES. For the first few iterations however, applying no preconditioner returns the smallest residual. The two remaining inexact constraint preconditioners both achieved similar levels of residual to taking $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, however they required a larger number of iterations to reach that level.

4.5.4 | Summary

In this section we have considered several preconditioner approaches for LR-GMRES applied to three different examples. In these examples we see that preconditioning is not as effective for LR-GMRES as in the GMRES examples in Section 4.3. This is largely due to the truncation steps within LR-GMRES. In these steps, only the most important modes are selected e.g. the ones belonging to larger eigenvalues, ignoring the smaller ones. Therefore, the low-rank approach loses some information.

The preconditioners which were most affected by this were the Schur complement preconditioners, for both the block diagonal and block triangular Schur complement preconditioners only small improvements in the residual were observed, and in all of the examples we considered here, the systems preconditioned using this approach did not achieve a smaller residual than when considering the unpreconditioned system.

In Section 4.3 the examples with partial observations required a larger number of iterations to converge than the equivalent examples with full observations. The opposite was true here, with the inexact constraint preconditioners being significantly more effective for the examples with partial observations. This suggests that the partial, and thus lower-rank observations improved the efficacy of the preconditioner, with the LR-GMRES method being able to exploit this. Taking a larger rank for the LR-GMRES method improved the performance of the inexact constraint preconditioners across majority of the examples, with this being more significant in the partial observations examples. In these particular examples the rank of the matrices within LR-GMRES and thus the tensor rank of the solution vector, was greater than the number of observations, and this may have contributed to the efficacy.

The most effective preconditioner of the ones considered here was the inexact constraint preconditioner with the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$. This resulted in the smallest residual from the preconditioned systems for the first 50 iterations in majority of examples, however as with the other preconditioners, it ultimately stagnated, and in some examples the remaining inexact constraint preconditioners achieved a lower residual.

In the partial observations examples, using the inexact constraint preconditioner with $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ outperformed not using a preconditioner after approximately 10 iterations for the $r = 20$ examples, and all but the advection-diffusion example taking $r = 5$. Indeed, not applying a preconditioner for LR-GMRES appeared to be a better choice than a large number of the preconditioners we considered here. This is likely due to the truncation process as mentioned above, with the low-rank approach acting as a form of regularisation and thus in some sense like a projected

preconditioner itself.

A possible approach for preconditioning the partial observations examples may be to use a "hybrid" approach, where no preconditioner is used for the first 10 to 20 iterations before applying the inexact constraint preconditioner with the approximations $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$.

In the next sections we shall consider including closer approximations to $\mathbf{L}$ and $\mathbf{L}^{-1}$ through truncation and the implementation of this approach.

4.6 | Truncating inverses in Kronecker form

Thus far we have presented preconditioners $\mathcal{P}$, where we apply the inverse $\mathcal{P}^{-1}$ to the matrix $\mathcal{A}$. In LR-GMRES we implement this through matrix multiplication for example Algorithm 4. When inverted these preconditioners have terms with $\tilde{\mathbf{L}}^{-1}$. For the approximations $\mathbf{I}$ and $\hat{\mathbf{L}}$ we considered in the above numerical results, the inverses $\tilde{\mathbf{L}}$ can each be written as one Kronecker product, making implementation within LR-GMRES easy.

In this section we consider approximations to $\mathbf{L}$ of the form $\tilde{\mathbf{L}} = I_{N+1} \otimes I_n + C \otimes \tilde{M}$ and consider the inverse:

$$\tilde{\mathbf{L}}^{-1} = \begin{bmatrix} I & & & \\ -\tilde{M} & I & & \\ & \ddots & \ddots & \\ & & -\tilde{M} & I \end{bmatrix}^{-1} = \begin{bmatrix} I & & & \\ \tilde{M} & I & & \\ \vdots & \ddots & \ddots & \\ \tilde{M}^N & \cdots & \tilde{M} & I \end{bmatrix}. \tag{4.14}$$

To write this in Kronecker form, we observe that the each diagonal can be written as $(-C)^k \otimes \tilde{M}^k$ since C is the matrix with -1 on the sub-diagonal. Thus when $(-C)$ is raised to each successive power we obtain the diagonal below, taking $-C$ to ensure the correct signs. Hence the inverse of $\tilde{\mathbf{L}}^{-1}$ can be written:

$$\begin{aligned} \tilde{\mathbf{L}}^{-1} &= I_{N+1} \otimes I_n - C \otimes \tilde{M} + C^2 \otimes \tilde{M}^2 - \ldots + C^N \otimes \tilde{M}^N \\ &= I_{N+1} \otimes I_n + \sum_{k=1}^{N} (-C)^k \otimes \tilde{M}^k. \end{aligned} \tag{4.15}$$

In order to tractably include more complicated approximations to $\mathbf{L}$ than those considered earlier in this chapter, we truncate the resulting inverse (4.15) and use this in our preconditioners.

When we include this truncated inverse into the inverse of our preconditioners, this results in an approximation $\mathcal{Q}$ to the actual inverse of the preconditioner $\mathcal{P}^{-1}$. We consider applying the preconditioner $\mathcal{Q}$ in the same approach as above, applying the matrix multiplication $\mathcal{A}\mathcal{Q}$ within GMRES or LR-GMRES. As with $\mathcal{P}^{-1}$, we do not need to form $\mathcal{Q}$ explicitly.

As an example, let us consider the inexact constraint preconditioner with $\tilde{\mathbf{L}}$ of the above form, and $\tilde{\mathbf{H}} = 0$:

$$\mathcal{P}^{-1} = \begin{bmatrix} \mathbf{D} & 0 & \tilde{\mathbf{L}} \\ 0 & \mathbf{R} & 0 \\ \tilde{\mathbf{L}}^T & 0 & 0 \end{bmatrix}^{-1} = \begin{bmatrix} 0 & 0 & \mathbf{L}^{-T} \\ 0 & \mathbf{R}^{-1} & 0 \\ \mathbf{L}^{-1} & 0 & -\tilde{\mathbf{L}}^{-1}\mathbf{D}\tilde{\mathbf{L}}^{-T} \end{bmatrix} \tag{4.16}$$

$$= \begin{bmatrix} 0 & 0 & (I \otimes I + C \otimes \tilde{M})^{-T} \\ 0 & I \otimes R^{-1} & 0 \\ (I \otimes I + C \otimes \tilde{M})^{-1} & 0 & \tilde{\mathbf{S}}^{-1} \end{bmatrix}, \tag{4.17}$$

if we truncate the sum in (4.15) after one term and substitute this into (4.17), we obtain:

$$\mathcal{P}^{-1} \approx \begin{bmatrix} 0 & 0 & I \otimes I - C \otimes \tilde{M} \\ 0 & I \otimes R^{-1} & 0 \\ I \otimes I - C \otimes \tilde{M} & 0 & \tilde{\mathbf{S}}_I^{-1} \end{bmatrix} = \mathcal{Q}, \tag{4.18}$$

where

$$\begin{aligned} \tilde{\mathbf{S}}^{-1} = &- (I \otimes I + C \otimes \tilde{M})^{-1}(E_1 \otimes B)(I \otimes I + C \otimes \tilde{M})^{-T} \\ &- (I \otimes I + C \otimes \tilde{M})^{-1}(E_2 \otimes Q)(I \otimes I + C \otimes \tilde{M})^{-T} \\ \approx &- (I \otimes I - C \otimes \tilde{M})(E_1 \otimes B)(I \otimes I - C^T \otimes \tilde{M}^T) \\ &- (I \otimes I - C \otimes \tilde{M})(E_2 \otimes Q)(I \otimes I - C^T \otimes \tilde{M}^T) = \tilde{\mathbf{S}}_I^{-1}. \end{aligned}$$

When applying these truncated inverses within the resulting approximated preconditioner in this way, the truncated inverse makes a significant difference to the efficacy. However a consideration must be made for the number of terms which are included in the approximation, with each additional term increasing the number of matrix vector products which must be applied within the solver.

In this section, we consider $\mathbf{L}$ and truncating the inverse as described above. This method applies only to the advection-diffusion, and shallow water equations examples from above, as these two methods have constant model matrices M and

thus the resulting $\mathbf{L}$ are of the form $I_{N+1} \otimes I_n + C \otimes M$ whilst the Lorenz example is not (we refer to Section 3.4).

We shall apply this approximation to the preconditioners introduced in Section 4.2 for both the data assimilation saddle point problem solved with GMRES, and with LR-GMRES.

4.6.1 | Numerical results for GMRES

Let us first consider the application of the truncated $\mathbf{L}^{-1}$ to the preconditioners using GMRES. To illustrate this approach we consider the advection-diffusion example from Section 4.3. In the figures to follow we present the three preconditioners separately with different levels of truncation. For comparison, we include the residuals obtained when not using a preconditioner, when taking the approximations $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, and for $\tilde{\mathbf{L}} = \mathbf{L}$ if we do not truncate at all. Furthermore we observe that if we do not include any terms from the sum in (4.15), we re-obtain the approximation $\tilde{\mathbf{L}} = \mathbf{I}$.

To illustrate the truncation of $\mathbf{L}^{-1}$ we consider here the partial ($p = 3$) observations example in Section 4.3.1 as there was the greatest disparity between taking the approximations $\tilde{\mathbf{L}} = \mathbf{I}$ and $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$. We consider the three preconditioners separately for clarity.

Diagonal Schur complement

First we present the results for truncating the inverse of $\mathbf{L}$ when using the block diagonal Schur complement preconditioner in Figure 4.25.

We observe that using the true inverse of $\mathbf{L}$ ($k = 29$), the preconditioner is very effective, with the residual reaching a level of 10^{-6} after only 25 iterations, despite not using the observation operator $\mathbf{H}$. In contrast, as we saw in Figure 4.14, using the approximation $\tilde{\mathbf{L}} = \mathbf{I}$ (or indeed truncating (4.15) at $k = 0$) is not very effective. Increasing k does increase the efficacy of the preconditioner, however it is only after taking $k = 20$ that the approximation to $\mathbf{L}^{-1}$ is more effective than taking the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$. We must note however than increasing k also increases the number of matrix vector products, and thus becomes more expensive to apply. As in Section 4.3.1, we see that with the exception of using the true inverse, not applying a preconditioner is more effective for the first 80 iterations.

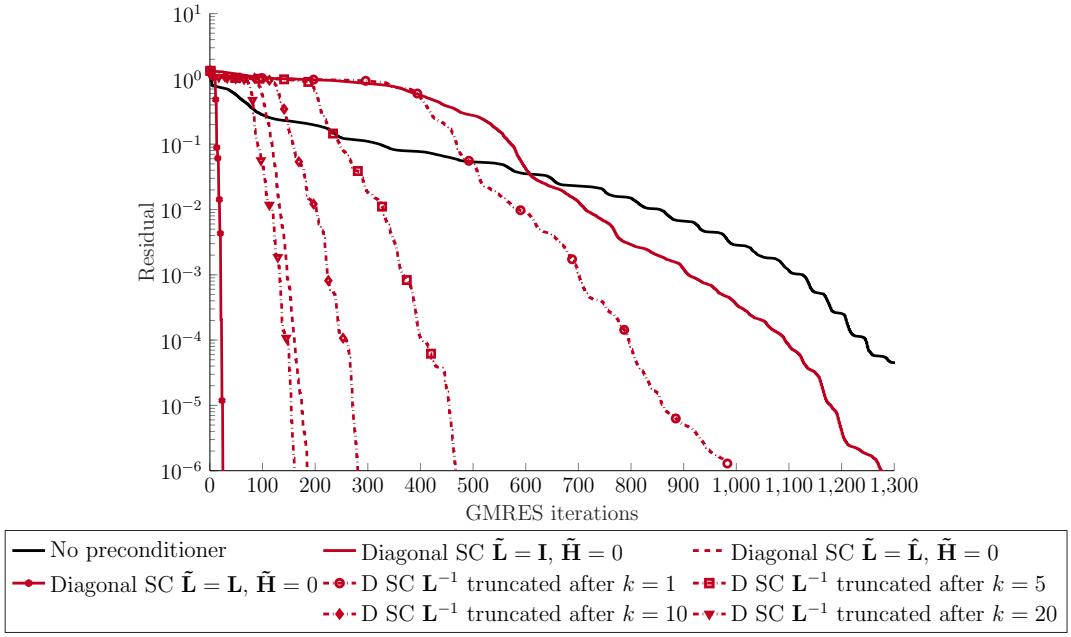


Figure 4.25: GMRES residual for the 1890×1890 advection-diffusion example with partial observations using block diagonal Schur complement preconditioners.

Triangular Schur complement

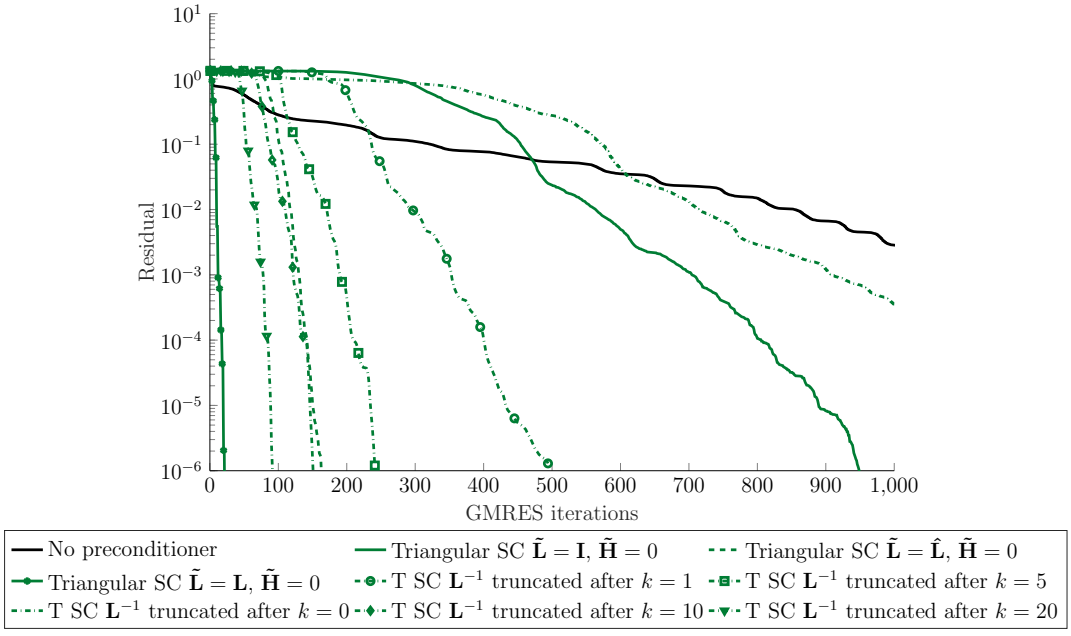


Figure 4.26: GMRES residual for the 1890×1890 advection-diffusion example with partial observations using block triangular Schur complement preconditioners.

When using the block triangular Schur complement preconditioners, we observe

that the inverted preconditioner contains a term with $\tilde{\mathbf{L}}$, hence truncating the sum in (4.15) with $\tilde{\mathbf{L}} = \mathbf{L}$ and $k = 0$ does not result in the same preconditioner as taking $\tilde{\mathbf{L}} = \mathbf{I}$. As seen in Figure 4.26 this performs even less effectively than using $\tilde{\mathbf{L}} = \mathbf{I}$ and only achieves a lower residual than the unpreconditioned system after 600 iterations.

We see that the inclusion of one additional term from (4.15) greatly increases the efficacy of the preconditioner, far more noticeably than for the block diagonal Schur complement preconditioner. Here we also observe that it is after only $k = 10$ terms that the truncated $\mathbf{L}^{-1}$ is more effective than taking the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, in contrast to the 20 needed in the block diagonal Schur complement example. As with the previous example we observe that initially these preconditioners when not using the true $\mathbf{L}^{-1}$, do not result in an improvement over not applying a preconditioner, and require significantly more matrix vector products.

Inexact constraint

In Section 4.3.1 we observed that the most effective of the preconditioners were the inexact constraint preconditioners, in Figure 4.27 we consider the effect of truncating (4.15) for these preconditioners.

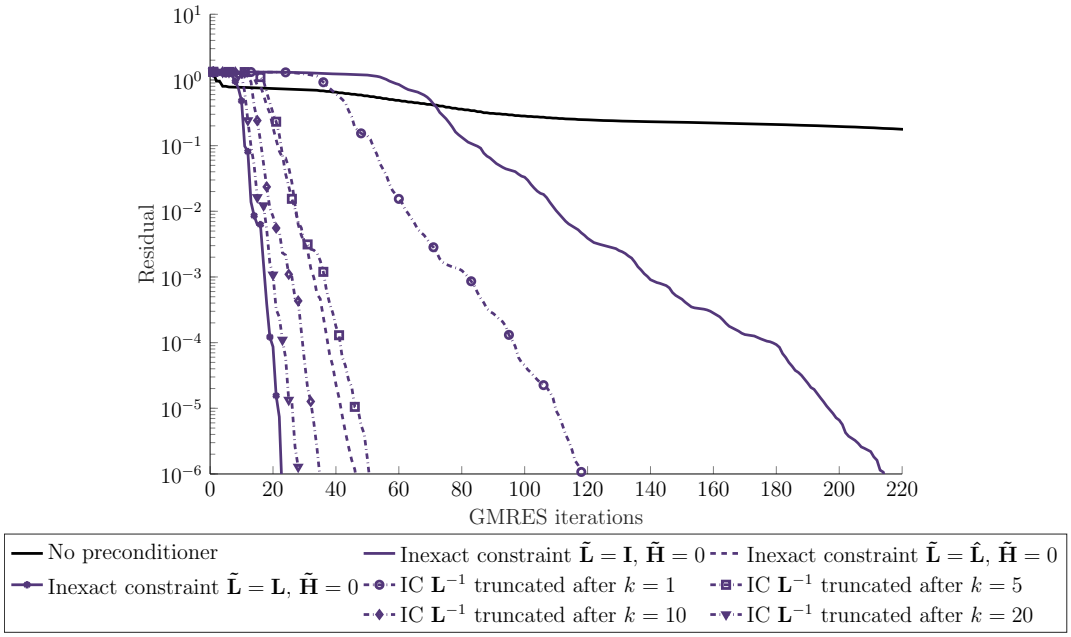


Figure 4.27: GMRES residual for the 1890×1890 advection-diffusion example with partial observations using inexact constraint preconditioners.

Here we observe that truncating (4.15) after only 5 terms results in a preconditioner which is similarly effective to the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, with the convergence

improving quite significantly for the first terms from that achieved with the approximation $\tilde{\mathbf{L}} = \mathbf{I}$. This is fewer terms than in the Schur complement preconditioners examples, however still requires more matrix vector products than the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$. Whilst the performance does improve for $k = 10$ and $k = 20$, getting closer to the efficacy of the preconditioner with the true $\mathbf{L}^{-1}$ it is more incremental improvements due to how effective the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ already is, and increases the number of matrix vector products further.

Summary

A common observation to all three preconditioners for the advection-diffusion example is that it takes a number of terms from (4.15) to achieve the efficacy of the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, and requires significantly more matrix vector products. This is particularly noticeable for the block diagonal Schur complement preconditioner in Figure 4.25 where it requires 20 terms from the sum in (4.15) to achieve the same convergence as simply approximating the model matrix M with the identity.

If we consider the shallow water equations example from Section 4.3.2 we note that the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ was very effective due to the closeness of the eigenvalues of the model matrix M to 1. Truncation of $\tilde{\mathbf{L}}^{-1}$ for this example would not be able to compete with the cheap approximation $M = I$, i.e. $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$.

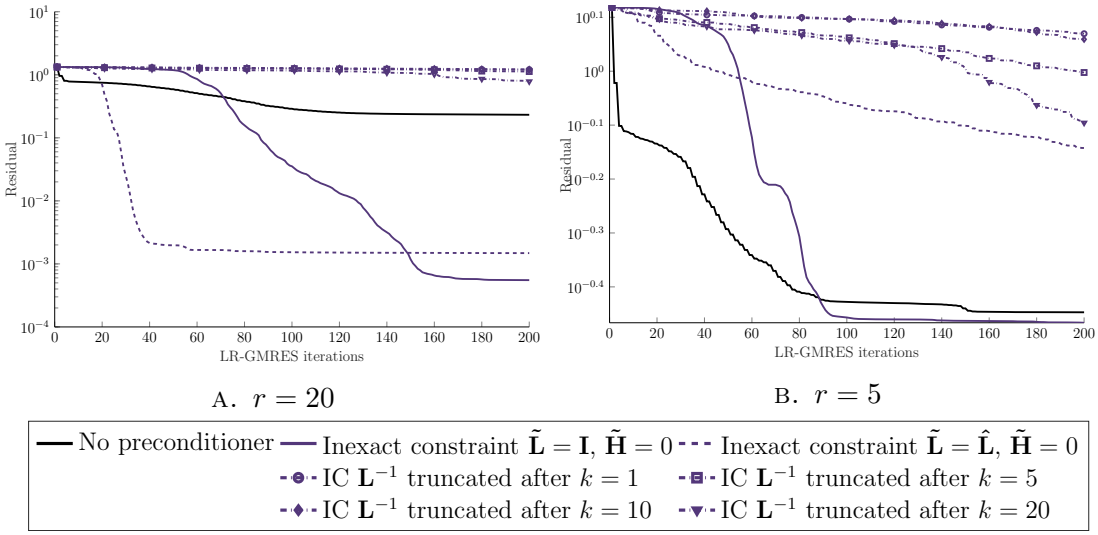
As such taking the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ is a more natural choice than truncating the inverse to $\mathbf{L}$, or another approximation of the form $\tilde{\mathbf{L}} = (I \otimes I + C \otimes \tilde{M})$.

4.6.2 | Low rank numerical results

Let us now apply these ideas to LR-GMRES, with the hope that including the model operator will lead to an improvement in the convergence of LR-GMRES. As with the GMRES example above, we consider the advection-diffusion example, here from Section 4.5.1 taking partial ($p = 3$) observations. We present only the inexact constraint preconditioner for this example, as we observed in Figure 4.20 that the Schur complement preconditioners were not effective for this problem.

Inexact constraint

We observe that as with applying GMRES, when truncating the sum in (4.15) at $k = 0$, we return to $\tilde{\mathbf{L}} = \mathbf{I}$. We consider the same levels of truncation $k = 0, 1, 5, 10, 20$ as in Section 4.6.1, and additionally the convergence using the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$.



A. $r = 20$ B. $r = 5$

Figure 4.28: LR-GMRES residual for the 1890×1890 advection-diffusion example with partial observations using inexact constraint preconditioners ($r = 20$, $r = 5$).

We observe here that the inclusion of a larger number of terms from (4.15) does improve the efficacy of the preconditioner which is more noticeable in the $r = 5$ example in Figure 4.28 B) due to the scale. Unfortunately including more terms does not achieve the same improvement we observed in Section 4.6.1, with level of truncating $\mathbf{L}^{-1}$ achieving the same level of efficacy as taking $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$, nor exceeding it.

This is likely due to the truncation inherent in the LR-GMRES algorithm. The additional terms from the approximation to $\mathbf{L}^{-1}$ results in additional concatenated matrices which must be truncated. As a result, some of the information which is gained by including the true model matrix is lost through this truncation step.

As we observed in Section 4.6.1, the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ is a cheap and effective choice for approximating $\mathbf{L}$. This is especially true when we consider the LR-GMRES method as the inverse is cheaper to consider, and does not lead to the same level of concatenation which is necessary when we consider approximations of the form $\tilde{\mathbf{L}} = (I \otimes I + C \otimes \tilde{M})$ or the true $\mathbf{L}$.

4.7 | Conclusions

In this chapter we have presented three different preconditioners and applied them to the data assimilation saddle point problem when solved with both GMRES and the low-rank GMRES method introduced in Chapter 3. In order to apply these preconditioners, we considered different approximations for the matrices $\mathbf{L}$ and $\mathbf{H}$.

We observed that when solving the data assimilation saddle point problem with GMRES, the most effective preconditioner was the inexact constraint preconditioner [16, 17, 18] taking the approximation $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ to the matrix $\mathbf{L}$, and approximating $\mathbf{H}$ by 0. In Section 4.6 we considered using the exact $\mathbf{L}$ and approximating its inverse, and whilst we did achieve superior results for close approximations to $\mathbf{L}^{-1}$, the evaluation of this is significantly more expensive. For LR-GMRES we observed that the inexact constraint preconditioner with $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ was also the most effective preconditioner for this method as well.

When considering the convergence of GMRES or LR-GMRES, we identified that whilst for GMRES having full observations each timestep led to faster convergence in comparison to taking partial observations, this was not true for LR-GMRES. In the low-rank setting we observed that for both the unpreconditioned system, and when using the inexact constraint preconditioners, the convergence was significantly improved for the examples where we considered partial observations. For the low-rank method, we investigated two different choices of rank, and in the numerical results we saw that taking a larger value for r generally led to better performance. A possible explanation for why preconditioning is not as effective majority of the time within LR-GMRES when compared to GMRES, and often leads to the convergence of LR-GMRES stagnating is the following. During LR-GMRES, the truncation process selects only the most important modes, e.g. the ones belonging to larger eigenvalues, ignoring the smaller ones. Therefore, the low-rank approach acts like a regularisation, and hence in some sense like a projected preconditioner.

The interesting observation to be made for both methods, and for majority of the examples we considered was that using no preconditioner returned the smallest residual for the first 10 to 20 iterations (and sometimes more). As a result we conclude that preconditioning the data assimilation saddle point problem may require further investigation into different types of preconditioners. A possible approach for preconditioning the low-rank method when considering partial observations may be to use a "hybrid" approach, where no preconditioner is used for the first 10 to 20 iterations before applying the inexact constraint preconditioner with the approximations $\tilde{\mathbf{L}} = \hat{\mathbf{L}}$ and $\tilde{\mathbf{H}} = 0$.

CHAPTER 5

PROJECTION METHODS FOR WEAK CONSTRAINT VARIATIONAL DATA ASSIMILATION

The work in this chapter is the basis of the paper [56] which has been submitted for publication.

5.1 | Introduction

When considering real world applications of data assimilation, such as numerical weather prediction, the dimension of the state space of these systems can become very large. As such, it is essential to consider projecting the variables onto a space of smaller dimension in order to solve a smaller problem which a) approximates the full-size problem effectively, and b) reduces the computational cost.

In this chapter we present projection methods for the weak constraint variational data assimilation problem. During the data assimilation minimisation, we must solve a linear system. Through projection methods, we are able to solve a significantly smaller system, reducing the complexity of this step. We extend the use of balanced truncation [96] from the strong constraint variational data assimilation case considered in [23, 24, 84, 85] to the weak constraint setting, and introduce randomised projection methods, sometimes known as sketching methods, to the data assimilation problem.

Model reduction methods have previously been considered within variational data assimilation, with papers considering balanced truncation, proper orthogonal decomposition (POD) and reduced basis methods, as well as low-rank approaches

as discussed in Chapter 2. In this chapter we take a different approach, making use of randomised algorithms which have seen growth in recent years [43, 67, 95]. There have been investigations into the efficacy of randomised approaches for matrix decompositions [67] and other numerical linear algebra techniques see for example [43]. Here we apply random projections, which have been used effectively for dimensionality reduction in other applications. For example, [20] uses these ideas for image and text data, whilst [89] applies sketching methods to inverse problems. There have been other applications of these randomised sketching methods to least squares problems and low-rank matrix approximation, see for example [103, 134] and references therein. Furthermore, we extend the application of balanced truncation from the strong constraint variational data assimilation setting considered in [23, 24, 84, 85] to the weak constraint setting. We compare the performance of these two approaches to the results obtained by solving the full-sized problem numerically, and with error analysis on the error introduced through projection.

In the remainder of this section we recall incremental weak constraint four dimensional variational data assimilation (4D-Var), introduced in greater detail in Chapter 3. Section 5.2 then introduces projected 4D-Var, before we introduce the balanced truncation and randomised methods as two special cases of projection type methods in Sections 5.3 and 5.4. The resulting error in the state space for the projected method compared to the full solution is presented in Section 5.5. Numerical results are given in Section 5.6.

Incremental 4D-Var

To implement 4D-Var operationally, an incremental approach [35] is used. This is essentially the Gauss-Newton method and generates an approximation to the solution of $x = \text{argmin } J(x)$. We approximate the 4D-Var cost function by a quadratic function of an increment

$$\delta x^{(\ell)} = x^{(\ell+1)} - x^{(\ell)}, \tag{5.1}$$

where $x^{(\ell)} = \left[(x_0^{(\ell)})^T, (x_1^{(\ell)})^T, \cdots, (x_N^{(\ell)})^T\right]^T$ denotes the ℓ-th iterate of the Gauss-Newton algorithm. This increment $\delta x^{(\ell)}$ is a solution to the minimisation of the

linearised cost function

$$\begin{aligned}\tilde{J}(\delta x^{(\ell)}) = &\frac{1}{2}\|\delta x_0^{(\ell)} - b_0^{(\ell)}\|^2_{B^{-1}} \\ &+ \frac{1}{2}\sum_{k=0}^{N}\|d_k^{(\ell)} - H_k\delta x_k^{(\ell)}\|^2_{R_k^{-1}} \\ &+ \frac{1}{2}\sum_{k=1}^{N}\|\delta x_k^{(\ell)} - M_k\delta x_{k-1}^{(\ell)} - c_k^{(\ell)}\|^2_{Q_k^{-1}},\end{aligned} \tag{5.2}$$

where $M_k \in \mathbb{R}^{n\times n}$ and $H_k \in \mathbb{R}^{n\times p_k}$, are linearisations of $\mathcal{M}_k$ and $\mathcal{H}_k$ about the current state trajectory $x^{(\ell)}$. Here we have used

$$b_0^{(\ell)} = x_0^b - x_0^{(\ell)}, \tag{5.3}$$

$$d_k^{(\ell)} = y_k - \mathcal{H}_k(x_k^{(\ell)}), \tag{5.4}$$

$$c_k^{(\ell)} = \mathcal{M}_k(x_{k-1}^{(\ell)}) - x_k^{(\ell)}. \tag{5.5}$$

Dropping the iterate (ℓ) for convenience, The cost function (5.2) can be written more concisely as

$$\tilde{J}(\delta x) = \frac{1}{2}\|\mathbf{L}\delta x - b\|^2_{D^{-1}} + \frac{1}{2}\|d - \mathbf{H}\delta x\|^2_{R_k^{-1}}, \tag{5.6}$$

where $\delta x = \left[\delta x_0^T, \delta x_1^T, \cdots, \delta x_N^T\right]^T \in \mathbb{R}^{(N+1)n}$ and $\mathbf{L}$ and $\mathbf{H}$ are matrices of size $(N+1)n \times (N+1)n$ and $\sum_{k=0}^{N} p_k \times (N+1)n$ respectively,

$$\mathbf{L} = \begin{bmatrix} I & & & \\ -M_1 & I & & \\ & \ddots & \ddots & \\ & & -M_N & I \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} H_0 & & & \\ & H_1 & & \\ & & \ddots & \\ & & & H_N \end{bmatrix} \tag{5.7}$$

which can be thought of as model and observation operators over the assimilation window.

We assume there is no correlation between timesteps, and hence the covariance matrices are block diagonal matrices

$$\mathbf{D} = \mathrm{diag}(B, Q_1, \cdots, Q_N), \quad \mathbf{R} = \mathrm{diag}(R_0, R_1, \cdots, R_N),$$

with corresponding vectors

$$b = \left[b_0^T, c_1^T, \cdots, c_N^T\right]^T \in \mathbb{R}^{(N+1)n}, \quad d = \left[d_0^T, d_1^T, \cdots, d_N^T\right]^T \in \mathbb{R}^{\sum_{k=0}^{N} p_k}.$$

Minimising $\tilde{J}(\delta x)$ results in solving a potentially very large linear system. In particular, the linearised model and observation operators M_k and H_k for each timestep can be very expensive to evaluate, and this motivates projecting the state space onto a subspace of smaller dimension. This is what we present in the remainder of this chapter.

5.2 | Projected weak constraint 4D-Var

We project M_k and H_k onto a lower dimensional space in order to perform the minimisation of (5.6) to find δx. We introduce a restriction operator $U^T \in \mathbb{R}^{r\times n}$ which maps the model variables δx_k to a lower-dimensional space, defining $\delta\hat{x}_k \in \mathbb{R}^r$, with $r \ll n$ as $\delta\hat{x}_k = U^T\delta x_k$. The minimisation problem will be carried out in a lower dimensional space of dimension $r \ll n$. Therefore we introduce the prolongation operator $V \in \mathbb{R}^{n\times r}$ which maps back to the original space, with the requirement that $U^TV = I_r$.

This allows us to define reduced model and observation operators $\hat{M}_k$ and $\hat{H}_k$ for the lower dimensional problem:

$$\begin{aligned}\hat{M}_k &= U^T M_k V \in \mathbb{R}^{r\times r},\\ \hat{H}_k &= H_k V \in \mathbb{R}^{p_k\times r}.\end{aligned}$$

The projection from δx to $\delta\hat{x}$ does not affect the observations, nor the observation error covariance matrices R_k. However the background error $b_0 = (x_0^b - x_0)$ becomes $\hat{b}_0 = U^T(x_0^b - x_0)$ and thus the covariance matrix B must be projected to $\hat{B} = U^TBU$. The same is true of the model error c_k, and corresponding covariance matrices Q_k, being projected here to $\hat{c}_k = U^Tc_k$ and $\hat{Q}_k = U^TQ_kU$ respectively.

The resulting linearised cost function for this reduced system is hence

$$\begin{aligned}\tilde{J}(\delta\hat{x}) = \frac{1}{2}\|\delta\hat{x}_0 - \hat{b}_0\|^2_{\hat{B}^{-1}} + \frac{1}{2}\sum_{k=0}^{N}\|d_k - \hat{H}_k\delta\hat{x}_k\|^2_{R_k^{-1}}\\ + \frac{1}{2}\sum_{k=1}^{N}\|\delta\hat{x}_k - \hat{M}_k\delta\hat{x}_{k-1} - \hat{c}_k\|^2_{\hat{Q}_k^{-1}},\end{aligned} \tag{5.8}$$

or indeed

$$\tilde{J}(\delta\hat{x}) = \frac{1}{2}\|\hat{\mathbf{L}}\delta\hat{x} - \hat{b}\|^2_{\hat{\mathbf{D}}^{-1}} + \frac{1}{2}\|d - \hat{\mathbf{H}}\delta\hat{x}\|^2_{\mathbf{R}^{-1}}, \tag{5.9}$$

which we wish to minimise.

Here $\hat{\mathbf{D}} = \mathbf{U}^T\mathbf{D}\mathbf{U}$ and $\hat{\mathbf{H}} = \mathbf{H}\mathbf{V}$, with $\hat{b} = \mathbf{U}^T b$ defining $\mathbf{U} = I \otimes U$ and $\mathbf{V} = I \otimes V$, whilst $\hat{\mathbf{L}} = \mathbf{U}^T\mathbf{L}\mathbf{V}$ due to the requirement that $U^TV = I_r$.

The choice of U and V will determine the efficacy of the method. We present two approaches in the remainder of this chapter. Firstly we consider balanced truncation, a control theoretic model reduction approach. Subsequently we introduce randomised methods for dimensionality reduction, and we compare both approaches to performing a coarsening grid method approach.

5.3 | Balanced truncation

Balanced truncation [96] is a model reduction method which has been applied within incremental 4D-Var for strong constraint variational data assimilation in [23, 24, 84, 85]. In order to apply balanced truncation within data assimilation, the system is linearised via the so-called tangent linear model.

Let us first consider some necessary concepts from control theory before introducing balanced truncation for linear time-invariant systems.

5.3.1 | Control theoretic preliminaries

The motivation for balanced truncation comes from the control theoretic desire to approximate the input-output map $u \to y$ of a linear discrete time-invariant system such as

$$\begin{aligned} x_{k+1} &= \mathbf{A}x_k + \mathbf{B}u_k, \\ y_k &= \mathbf{C}x_k, \end{aligned} \tag{5.10}$$

where at each timestep k we have a state $x_k \in \mathbb{R}^n$, input $u_k \in \mathbb{R}^m$ and output $y_k \in \mathbb{R}^p$. The matrices $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$ are time-invariant, and of sizes $n \times n, n \times m$ and $p \times n$ respectively.

In balanced truncation, the system is transformed such that the states x_k in (5.10) which are difficult to reach, and those which are difficult to observe coincide. The states which are most difficult are eliminated, thus obtaining a reduced order

system:

$$\begin{aligned} \hat{x}_{k+1} &= \hat{\mathbf{A}}\hat{x}_k + \hat{\mathbf{B}}u_k, \\ \hat{y}_k &= \hat{\mathbf{C}}\hat{x}_k, \end{aligned} \tag{5.11}$$

where $\hat{\mathbf{A}} \in \mathbb{R}^{r\times r}, \hat{\mathbf{B}} \in \mathbb{R}^{r\times m}$ and $\hat{\mathbf{C}} \in \mathbb{R}^{p\times r}$, with $r \ll n$ which approximates (5.10).

Reachability and observability are important concepts in control theory which are defined as follows:

Definition (Reachability). *A state is* reachable *from a zero initial state if there exists an input function of finite energy such that the state is reached in a finite time interval.*

The linear discrete time-invariant system (5.10) *is* reachable *if all states $x \in \mathbb{R}^n$ are reachable.*

Definition (Observability). *A state is* observable *if when considered as an initial state, it can be determined from the system output that has been observed within a finite time interval.*

The linear discrete time-invariant system (5.10) *is* observable *if all states $x \in \mathbb{R}^n$ are observable.*

For the remainder of this chapter we assume that the linear time-invariant systems we consider are both reachable and observable.

In order to consider the states which are *difficult* to reach and observe, we must have a notion of the energy needed to reach or observe a state. We consider the infinite reachability and observability Gramians $\mathcal{G}_r$ and $\mathcal{G}_o$ of the system, which are defined only for stable systems.

Definition (Stability). *A linear discrete time-invariant system* (5.10) *is* stable *if all eigenvalues λ of $\mathbf{A}$ lie inside the unit disk: $|\lambda| < 1$ for all $\lambda \in \sigma(\mathbf{A})$.*

Definition (Infinite Gramians). *Let the linear discrete time-invariant system* (5.10) *be stable, reachable and observable. The infinite reachability and observability Gramians of the system* (5.10)*, $\mathcal{G}_r$ and $\mathcal{G}_o$ are*

$$\mathcal{G}_r = \sum_{j=0}^{\infty} \mathbf{A}^j \mathbf{B}\mathbf{B}^T (\mathbf{A}^T)^j, \tag{5.12}$$

$$\mathcal{G}_o = \sum_{j=0}^{\infty} (\mathbf{A}^T)^j \mathbf{C}^T \mathbf{C}\mathbf{A}^j, \tag{5.13}$$

respectively.

Lemma 5.1. *The infinite reachability and observability Gramians $\mathcal{G}_r$ and $\mathcal{G}_o$ satisfy the Stein (or discrete Lyapunov) equations:*

$$\mathcal{G}_r = \mathbf{A}\mathcal{G}_r\mathbf{A}^T + \mathbf{B}\mathbf{B}^T, \tag{5.14}$$

$$\mathcal{G}_o = \mathbf{A}^T\mathcal{G}_o\mathbf{A} + \mathbf{C}^T\mathbf{C}. \tag{5.15}$$

Proof. We consider the right hand side of (5.14),

$$\begin{aligned}\mathbf{A}\mathcal{G}_r\mathbf{A}^T + \mathbf{B}\mathbf{B}^T &= \sum_{j=0}^{j=\infty} \mathbf{A}^{j+1}\mathbf{B}\mathbf{B}^T(\mathbf{A}^T)^{j+1} + \mathbf{B}\mathbf{B}^T \\ &= \sum_{j=1}^{j=\infty} \mathbf{A}^{j}\mathbf{B}\mathbf{B}^T(\mathbf{A}^T)^{j} + \mathbf{A}^{0}\mathbf{B}\mathbf{B}^T(\mathbf{A}^T)^{0} \\ &= \sum_{j=0}^{j=\infty} \mathbf{A}^{j}\mathbf{B}\mathbf{B}^T(\mathbf{A}^T)^{j} = \mathcal{G}_r.\end{aligned}$$

The same method can be used for the Stein equation for the observability Gramian (5.15). □

Solving the Stein equations (5.14) and (5.15) provide a computable way of obtaining the Gramians. In practice this is computationally expensive and as a result iterative solvers such as low-rank Smith methods [87], Krylov subspace methods [73] and combinations of these approaches [11, 112] are used. These adapt the Smith method [121] and compute approximate solutions $\tilde{\mathcal{G}}_r = Z_r Z_r^T$ to the Stein equation (5.14) obtaining the low-rank factor Z_r, and equivalently for (5.15). We refer to [118] for further discussion on this topic. As we shall see when applying balanced truncation, it is often convenient to obtain a factor Z_r of the Gramian.

The following lemma provides a way to determine the states which are the most difficult to reach and to observe using the infinite Gramians defined above.

Lemma 5.2. *[4] The minimal energy required to reach the state x from an initial state of* 0 *is*

$$x^T\mathcal{G}_r^{-1}x.$$

The maximal energy produced by observing the output of the system whose initial state is x is

$$x^T\mathcal{G}_o x.$$

Hence the states which are most difficult, i.e. those which require the most energy to reach, are in the span of the eigenvectors of $\mathcal{G}_r$ corresponding to the smallest eigenvalues. Equivalently, the states which are difficult to observe, i.e. those

producing the smallest observation energy, are in the span of the eigenvectors of $\mathcal{G}_o$ corresponding to small eigenvalues. Thus if we desire to have a system where states x_k which are difficult to reach are also difficult to observe, we wish the Gramians $\mathcal{G}_r$ and $\mathcal{G}_o$ to be equal.

The description (5.10) is not unique, if we apply a state space transformation to the system (5.10), taking $x = T\hat{x}$, we obtain the equivalent linear time-invariant system

$$\begin{aligned} \hat{x}_{k+1} &= T^{-1}\mathbf{A}T\hat{x}_k + T^{-1}\mathbf{B}u_k, \\ y_k &= \mathbf{C}Tx_k, \end{aligned} \tag{5.16}$$

with the input-output behaviour of this system remaining affected by this transformation.

The Stein equations for the transformed discrete linear time-invariant system (5.16) are:

$$\hat{\mathcal{G}}_r = T^{-1}\mathbf{A}T\hat{\mathcal{G}}_r T^T\mathbf{A}^T T^{-T} + T^{-1}\mathbf{B}\mathbf{B}^T T^{-T}, \tag{5.17}$$

$$\hat{\mathcal{G}}_o = T^T\mathbf{A}^T T^{-T}\hat{\mathcal{G}}_o T^{-1}\mathbf{A}T + T^T\mathbf{C}^T\mathbf{C}T, \tag{5.18}$$

and hence the transformed Gramians are $\hat{\mathcal{G}}_o = T^T\mathcal{G}_o T$ and $\hat{\mathcal{G}}_r = T^{-1}\mathcal{G}_r T^{-T}$.

The product of these transformed Gramians $\hat{\mathcal{G}}_o\hat{\mathcal{G}}_r$, and the product of the Gramians from the original system $\mathcal{G}_o\mathcal{G}_r$ are related by a similarity transformation:

$$\mathcal{G}_o\mathcal{G}_r = T^T\mathcal{G}_o T T^{-1}\mathcal{G}_r T^{-T} = T^T\mathcal{G}_o\mathcal{G}_r T^{-T}.$$

Thus $\mathcal{G}_o\mathcal{G}_r$ and $\hat{\mathcal{G}}_o\hat{\mathcal{G}}_r$ have the same eigenvalues. The positive square roots of these eigenvalues are known as the Hankel singular values of the system, an important system invariant.

Performing a transformation T such that $\hat{\mathcal{G}}_r = \hat{\mathcal{G}}_o$ is known as balancing, with the resulting system being called balanced. From a balanced system, we may truncate the system, removing those states which are both difficult to reach and observe. The combination of these two steps to create a reduced order model is the model reduction method balanced truncation.

5.3.2 | Balanced truncation for discrete linear time-invariant systems

There are several possibilities to obtain a transformation for implementing balanced truncation c.f. [4]. We illustrate one approach below.

Let $\mathcal{G}_r = KK^T$, $\mathcal{G}_o = LL^T$ be decompositions of the respective Gramians. As noted above, K and L are usually computed directly from the discrete Lyapunov equations (5.14),(5.15) rather than decomposing the Gramians. We compute the singular value decomposition

$$K^T L = Z\Sigma Y^T, \tag{5.19}$$

where $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n)$ are the Hankel singular values as introduced above. We define the following projection matrices which result in a balanced, truncated system where the Gramians are equal, and the system is truncated via a rank-r approximation:

$$V = KZ_r\Sigma_r^{-\frac{1}{2}} \in \mathbb{R}^{n\times r}, \tag{5.20}$$

$$U = LY_r\Sigma_r^{-\frac{1}{2}} \in \mathbb{R}^{n\times r}. \tag{5.21}$$

Here Z_r and Y_r are the first r columns of Z and Y respectively, and the Hankel singular values which are kept are $\Sigma_r = \text{diag}(\sigma_1, \ldots, \sigma_r)$.

Letting $T = V$, we see U^T is the left inverse of T:

$$\begin{aligned} U^TV &= \Sigma_r^{-\frac{1}{2}}Y_r^TL^TKZ_r\Sigma_r^{-\frac{1}{2}} \\ &= \Sigma_r^{-\frac{1}{2}}Y_r^TY\Sigma Z^TZ_r\Sigma_r^{-\frac{1}{2}} \\ &= \Sigma_r^{-\frac{1}{2}}[I_r \quad 0]\Sigma[I_r \quad 0]^T\Sigma_r^{-\frac{1}{2}} = I_r. \end{aligned}$$

Thus V and U satisfy $U^TV = I_r$, however neither are orthogonal.

Applying the transformation $T = V$ to the discrete linear time-invariant system (5.10), the Gramians of the transformed system are equal. Taking the infinite reachability Gramian first we see

$$\begin{aligned} \hat{\mathcal{G}}_r &= U^T\mathcal{G}_rU \\ &= \Sigma_r^{-\frac{1}{2}}Y_r^TL^T\mathcal{G}_rLY_r\Sigma_r^{-\frac{1}{2}} \\ &= \Sigma_r^{-\frac{1}{2}}[I_r \quad 0]\Sigma^2[I_r \quad 0]^T\Sigma_r^{-\frac{1}{2}} \\ &= \Sigma_r, \end{aligned}$$

and similarly for the infinite observability Gramian,

$$\begin{aligned}\hat{\mathcal{G}}_o &= V^T \mathcal{G}_o V \\ &= \Sigma_r^{-\frac{1}{2}} Z_r^T K^T \mathcal{G}_o K Z_r \Sigma_r^{-\frac{1}{2}} \\ &= \Sigma_r^{-\frac{1}{2}} Z_r^T K^T L L^T K Z_r \Sigma_r^{-\frac{1}{2}} \\ &= \Sigma_r^{-\frac{1}{2}} Z_r^T Z \Sigma Y^T Y \Sigma Z^T Z_r \Sigma_r^{-\frac{1}{2}} \\ &= \Sigma_r^{-\frac{1}{2}} [I_r \quad 0] \Sigma^2 [I_r \quad 0]^T \Sigma_r^{-\frac{1}{2}} \\ &= \Sigma_r.\end{aligned}$$

Hence the system obtained by applying this transformation is balanced.

Limitations of applying balanced truncation

Limitations of balanced truncation for model reduction are the time-invariance of the system, and that the matrix $\mathbf{A}$ in (5.10) must be stable, which in the discrete setting, corresponds to the spectrum of $\mathbf{A}$ contained within the unit ball. The infinite Gramians (5.12) and (5.13) are defined only for stable systems, and are necessary to consider for the system to be balanced. There have been extensions to balanced truncation which aim to overcome these issues, such as [79, 80, 113, 116] and the references therein for time varying systems. These extensions have included using a data-based approach similar to POD, and considering time-varying Gramians. For unstable systems there have been proposals which split the system into stable and unstable parts, or shift the system, see [8, 46, 143, 144] and references within. Furthermore in [23], an alternative approach of scaling the system matrices has been considered.

One further limiting factor to the efficacy of balanced truncation within data assimilation, is the cost involved, notably solving the Stein equations (5.15), (5.14) to obtain the Gramians, and computing the singular value decomposition in (5.19). When balanced truncation is applied in other settings, the cost of the model reduction is amortised by reusing the same reduced model over multiple applications of the reduced system. However in data assimilation, each assimilation (typically) leads to a new system which must be then reduced. Hence the cost is freshly incurred each time, unless a linear time-invariant system is considered. In the remainder of this section we consider applying balanced truncation within weak constraint variational data assimilation, and in Section 5.6 shall see the efficacy of such a reduced system.

5.3.3 | Balanced truncation within the weak constraint 4D-Var method

The authors of [23, 24, 84, 85] apply a modified version of balanced truncation within incremental strong constraint 4D-Var. Here we consider applying a similar approach to weak constraint 4D-Var, and a time-invariant system. Let us assume that the model and observation operators are time-invariant, with $M_k = M$ and $H_k = H$ for all k. The tangent linear model in the inner loop of incremental 4D-Var is considered as the linear discrete stochastic system

$$\begin{aligned} \delta x_{-1} &= 0, \\ \delta x_{k+1} &= M\delta x_k + u_k, \\ d_k &= H\delta x_k, \end{aligned} \tag{5.22}$$

where, in the weak constraint data assimilation case, the inputs are:

$$u_k = \begin{cases} e_0 \sim \mathcal{N}(0, B), & \text{for } k = -1 \\ \eta_k \sim \mathcal{N}(0, Q), & \text{for } k \geq 0. \end{cases} \tag{5.23}$$

Here we make the further assumption that the model, and observation error covariances are time-invariant, $Q_k = Q, R_k = R$ for all k.

The resulting infinite reachability and observability Gramians $\mathcal{G}_r$ and $\mathcal{G}_o$ of the system (5.22) are

$$\mathcal{G}_r = \sum_{j=1}^{\infty} M^j Q (M^T)^j + B, \tag{5.24}$$

$$\mathcal{G}_o = \sum_{j=0}^{\infty} (M^T)^j H^T R H M^j. \tag{5.25}$$

These follow from the conditions described in [19] and [23].

As seen with the discrete linear time-invariant system (5.10) in Lemma 5.1, the Gramians (5.24) and (5.25) are the solutions to Stein equations.

Lemma 5.3. *The infinite reachability and observability Gramians $\mathcal{G}_r$ and $\mathcal{G}_o$ satisfy the Stein (or discrete Lyapunov) equations:*

$$\mathcal{G}_r = M\mathcal{G}_r M^T + B + M(Q - B)M^T, \tag{5.26}$$

$$\mathcal{G}_o = M^T \mathcal{G}_o M + H^T R H. \tag{5.27}$$

Proof. The observability Gramian is as in Lemma 5.1:

$$\begin{aligned} M^T\mathcal{G}_oM + H^TRH &= \sum_{j=0}^{\infty}(M^T)^{j+1}H^TRHM^{j+1} + H^TRH \\ &= \sum_{j=1}^{\infty}(M^T)^jH^TRHM^j + (M^T)^0H^TRHM^0 \\ &= \sum_{j=0}^{\infty}(M^T)^jH^TRHM^j = \mathcal{G}_o. \end{aligned}$$

For the reachability Gramian, we consider the right hand side of (5.26) as before,

$$\begin{aligned} M\mathcal{G}_rM^T + B + M(Q-B)M^T &= \sum_{j=1}^{\infty}M^{j+1}Q(M^T)^{j+1} \\ &\quad + MBM^T + B + M(Q-B)M^T \\ &= \sum_{j=2}^{\infty}M^jQ(M^T)^j + B + MQM^T \\ &= \sum_{j=1}^{\infty}M^jQ(M^T)^j + B = \mathcal{G}_r. \end{aligned}$$

□

As introduced above, there are multiple transformations for implementing balanced truncation. We can decompose $\mathcal{G}_r = KK^T$ and $\mathcal{G}_o = LL^T$ as before despite the different discrete Lyapunov equations. Proceeding in the same manner, we compute the singular value decomposition $K^TL = Z\Sigma Y^T$, and define the projection matrices:

$$V = KZ_r\Sigma_r^{-\frac{1}{2}} \in \mathbb{R}^{n\times r}, \tag{5.28}$$

$$U = LY_r\Sigma_r^{-\frac{1}{2}} \in \mathbb{R}^{n\times r}. \tag{5.29}$$

Here as before, Z_r and Y_r are the first r columns of Z and Y respectively, and the Hankel singular values which are kept are $\Sigma_r = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$.

α-bounded balanced truncation

As mentioned at the start of Section 5.3, balanced truncation applies only to stable time-invariant systems, where the spectrum of the model matrix (in our setting M) is within the unit ball. In order to overcome the stability requirement of the

system, the author in [23] introduces α-bounded balanced truncation in which the discrete linear time-invariant system is shifted. Let α be such that the eigenvalues of M are bounded by a disk of radius α from the origin. The balanced truncation method is applied to the shifted system

$$\begin{aligned} \delta x_{k+1} &= M_\alpha \delta x_k + u_k, \\ d_k &= H_\alpha \delta x_k, \end{aligned}$$

with corresponding background covariance matrix B_α, where

$$M_\alpha = M/\alpha, \quad H_\alpha = H/\sqrt{\alpha}, \quad B_\alpha = B/\sqrt{\alpha}.$$

This system is a stable discrete linear time-invariant system, on which balanced truncation can be applied to obtain the projection matrices U and V. This provides a method to apply balanced truncation to unstable systems. We refer to [23] for further details. In our numerical experiments in Section 5.6 we apply this approach to the unstable example of the shallow water equations.

5.4 | Randomised methods

In this section, we introduce randomised methods for projection. In contrast to the previous section where we used the balanced truncation method to compute the projection matrices U and V, here we wish to generate these projection matrices U and V from a random distribution. Constructing the choice of U and V in balanced truncation is an expensive step requiring the solution to two Stein equations. Thus generating random matrices U and V provides a significantly cheaper way for obtaining projection matrices.

Random methods for dimensionality reduction have been previously used for image data [20], inverse problems [89] and other applications, see references within. The motivation behind randomised methods for dimensionality reduction comes from the Johnson - Lindenstrauss (JL) Lemma [74] which says that when points are projected to a (sufficiently large) random subspace, the distances between them are approximately preserved.

Theorem 5.4 (Johnson-Lindenstrauss Lemma). *For any $0 < \epsilon < \frac{1}{2}$, let $V \subset \mathbb{R}^d$ be a set of n points and $k = 20\epsilon^{-2}\log(n)$. Then there exists a map $f\colon \mathbb{R}^d \to \mathbb{R}^k$ such*

that for all $u, v \in V$,

$$(1-\epsilon)\|u-v\|^2 \leq \|f(u)-f(v)\|^2 \leq (1+\epsilon)\|u-v\|^2. \tag{5.30}$$

For a proof we refer to [38].

Another name used for these methods is sketching, as the resulting matrix, once projected, is a 'sketch' of the original. Sketching methods can be considered as projection methods or sampling methods. The sampling based methods are data-dependent, and can potentially be quite expensive if considering an importance based sampling method, where the probability of sampling a column is dependent on a weighted norm of that column.

In this chapter we consider projection methods rather than sampling based methods, and take U^T to be the Moore-Penrose pseudoinverse of V, such that $U^T V$ approximates I_r.

Projection matrices from a distribution

In previous papers about randomised dimension reduction [38, 89, 95], and references within, taking a normally distributed random matrix has worked very effectively, and taking an approximation for the *Gaussian distribution* [1, 20], this can be generated efficiently. Here we consider

$$V = \sqrt{\frac{n}{r}}G, \qquad G \in \mathbb{R}^{n\times r} \sim \mathcal{N}(0, B), \tag{5.31}$$

where each column of G is drawn from a multivariate normal distribution zero mean with covariance B, our background covariance matrix.

An alternative approach is to consider a matrix where the entries are *uniformly distributed* taking

$$V = \sqrt{\frac{n}{r}}G, \qquad G \in \mathbb{R}^{n\times r} \sim \mathrm{Uni}(0,1). \tag{5.32}$$

Our last approach, as used in [1] is taking a matrix V where each entry of G is drawn from a *Rademacher distribution*:

$$V = \sqrt{\frac{n}{r}}G, \qquad G_{ij} = \begin{cases} +1 & \text{with probability } 1/2, \\ -1 & \text{with probability } 1/2. \end{cases} \tag{5.33}$$

By the JL Lemma, the expected norm of a projection of a unit vector onto a random subspace through the origin is $\sqrt{\frac{r}{n}}$, as such we scale our projection matrices

by $\sqrt{\frac{n}{r}}$.

All three of the random matrices we consider are dense matrices, with all non-zero entries. In Section 5.6, we compare the efficacy of these projection methods with a sparse coarse interpolatory matrix of the same size $n \times r$. Some of the other approaches which we do not consider in the following numerical examples are: CountSketch, a data streaming inspired method which works effectively for large sparse data, statistical leverage score sampling methods, the fast Johnson-Lindenstrauss transform, and the Nyström method. The latter requiring that the matrix or matrices we are projecting are symmetric positive semidefinite, which in general is not the case for our model matrix M.

5.5 | Projection error

In this section, we compare the resulting update vectors from solving the cost function for the full system and the projected system, (5.6) and (5.9).

Taking the gradient of (5.6) we can write the solution to the full state system as

$$\delta x = (\mathbf{L}^T\mathbf{D}^{-1}\mathbf{L} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}(\mathbf{L}^T\mathbf{D}^{-1}b + \mathbf{H}^T\mathbf{R}^{-1}d), \tag{5.34}$$

and the corresponding solution to the projected problem using (5.9) is

$$\delta\hat{x} = (\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\hat{\mathbf{L}} + \hat{\mathbf{H}}^T\mathbf{R}^{-1}\hat{\mathbf{H}})^{-1}(\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\hat{b} + \hat{\mathbf{H}}^T\mathbf{R}^{-1}d). \tag{5.35}$$

We are interested in the error between these two state vector updates, to compare the errors, we need to project $\delta\hat{x}$ back to the original size: $\|\delta x - \mathbf{V}\delta\hat{x}\|$. Using

$$\begin{aligned}\mathbf{S} &= (\mathbf{L}^T\mathbf{D}^{-1}\mathbf{L} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}),\\ \hat{\mathbf{S}} &= (\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\hat{\mathbf{L}} + \hat{\mathbf{H}}^T\mathbf{R}^{-1}\hat{\mathbf{H}}),\end{aligned}$$

noting that these are indeed (minus) the Schur complements of the saddle point formulation of weak constraint 4D-Var, we manipulate (5.34) and (5.35) to obtain

$$\begin{aligned}\delta x - \mathbf{V}\delta\hat{x} = \mathbf{S}^{-1}\mathbf{L}^T\mathbf{D}^{-1}b - \mathbf{V}\hat{\mathbf{S}}^{-1}\mathbf{U}^T b\\ + \mathbf{S}^{-1}\mathbf{H}^T\mathbf{R}^{-1}d - \mathbf{V}\hat{\mathbf{S}}^{-1}\hat{\mathbf{H}}^T\mathbf{R}^{-1}d.\end{aligned} \tag{5.36}$$

We consider the background and model error terms, those containing b first:

$$\begin{aligned}\mathbf{S}^{-1}\mathbf{L}^T\mathbf{D}^{-1}b - \mathbf{V}\hat{\mathbf{S}}^{-1}\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\mathbf{U}^T b = \mathbf{S}^{-1}[&\mathbf{L}^T\mathbf{D}^{-1}b \\ &- \mathbf{S}\mathbf{V}(\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T})\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\mathbf{U}^T b \\ &+ \mathbf{S}\mathbf{V}(\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T}\hat{\mathbf{H}}^T\hat{\mathbf{F}}^{-1}\hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T})\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\mathbf{U}^T b].\end{aligned}$$

Here we have made use of the Sherman-Morrison- Woodbury formula to rewrite the inverse of $\hat{\mathbf{S}}$ e.g.

$$\hat{\mathbf{S}}^{-1} = \hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T} - \hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T}\hat{\mathbf{H}}^T\hat{\mathbf{F}}^{-1}\hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T},$$

where $\hat{\mathbf{F}} = (\mathbf{R} + \hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T}\hat{\mathbf{H}}^T)$. We now substitute $\mathbf{S}$, obtaining:

$$\begin{aligned}\mathbf{S}^{-1}\mathbf{L}^T\mathbf{D}^{-1}b - \mathbf{V}\hat{\mathbf{S}}^{-1}\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\mathbf{U}^T b = \mathbf{S}^{-1}[&\mathbf{L}^T\mathbf{D}^{-1}b \\ &- \mathbf{L}^T\mathbf{D}^{-1}\mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\mathbf{U}^T b \\ &+ \mathbf{L}^T\mathbf{D}^{-1}\mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T}\hat{\mathbf{H}}^T\hat{\mathbf{F}}^{-1}\hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\mathbf{U}^T b \\ &+ \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{V}\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T}\hat{\mathbf{H}}^T\hat{\mathbf{F}}^{-1}\hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\mathbf{U}^T b \\ &+ \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{V}\hat{\mathbf{L}}^{-1}\mathbf{U}^T b].\end{aligned}$$

Adding and subtracting the term $\mathbf{H}^T\hat{\mathbf{F}}^{-1}\mathbf{H}\mathbf{L}^{-1}\mathbf{U}^T b$, the resulting expression is

$$\begin{aligned}\mathbf{S}^{-1}\mathbf{L}^T\mathbf{D}^{-1}b - \mathbf{V}\hat{\mathbf{S}}^{-1}\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\mathbf{U}^T b = \mathbf{S}^{-1}[&\mathbf{L}^T\mathbf{D}^{-1}(\mathbf{I} - \mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\mathbf{U}^T)b \\ &+ (\mathbf{L}^T\mathbf{D}^{-1}\mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T}\hat{\mathbf{H}}^T - \mathbf{H}^T)\hat{\mathbf{F}}^{-1}\hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\mathbf{U}^T b \\ &+ \mathbf{H}^T(\mathbf{R}^{-1}(\mathbf{R} + \mathbf{H}\mathbf{V}\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T}\hat{\mathbf{H}}^T)\hat{\mathbf{F}}^{-1}\hat{\mathbf{H}})\hat{\mathbf{L}}^{-1}\mathbf{U}^T b \\ &- \mathbf{H}^T(\mathbf{R}^{-1}\mathbf{H}\mathbf{V})\hat{\mathbf{L}}^{-1}\mathbf{U}^T b].\end{aligned}$$

However we notice that $\mathbf{H}\mathbf{V} = \hat{\mathbf{H}}$, and as such this simplifies to

$$\begin{aligned}\mathbf{S}^{-1}\mathbf{L}^T\mathbf{D}^{-1}b - \mathbf{V}\hat{\mathbf{S}}^{-1}\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\mathbf{U}^T b = &\mathbf{S}^{-1}\mathbf{L}^T\mathbf{D}^{-1}(\mathbf{I} - \mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\mathbf{U}^T)b \\ &- \mathbf{S}^{-1}\mathbf{J}\mathbf{H}^T\hat{\mathbf{F}}^{-1}\hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\mathbf{U}^T b,\end{aligned}$$

where $\mathbf{J} = (\mathbf{I} - \mathbf{L}^T\mathbf{D}^{-1}\mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T}\mathbf{V}^T)$.

Taking the same approach for the observation error terms, we obtain

$$\begin{aligned}\mathbf{S}^{-1}\mathbf{H}^T\mathbf{R}^{-1}d - \mathbf{V}\hat{\mathbf{S}}^{-1}\hat{\mathbf{H}}^T\mathbf{R}^{-1}d &= \mathbf{S}^{-1}\mathbf{J}\mathbf{H}^T\mathbf{R}^{-1}d \\ &\quad - \mathbf{S}^{-1}\mathbf{J}\mathbf{H}^T\hat{\mathbf{F}}^{-1}\hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\hat{\mathbf{D}}\hat{\mathbf{L}}^{-T}\hat{\mathbf{H}}^T\mathbf{R}^{-1}d \\ &= \mathbf{S}^{-1}\mathbf{J}\mathbf{H}^T\hat{\mathbf{F}}^{-1}d.\end{aligned}$$

Thus, returning to (5.36), we have

$$\begin{aligned}\delta x - \mathbf{V}\delta\hat{x} = \mathbf{S}^{-1}\mathbf{L}^T\mathbf{D}^{-1}(\mathbf{I} - \mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\mathbf{U}^T)b \\ + \mathbf{S}^{-1}\mathbf{J}\mathbf{H}^T\hat{\mathbf{F}}^{-1}f,\end{aligned}$$

where $f = (d - \hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\mathbf{U}^T b)$. Manipulating the second term, we obtain

$$\begin{aligned}\delta x - \mathbf{V}\delta\hat{x} = \mathbf{S}^{-1}(\mathbf{I} - \mathbf{L}^T\mathbf{U}\hat{\mathbf{L}}^{-T}\mathbf{V}^T)\mathbf{H}^T\hat{\mathbf{F}}^{-1}f \\ + \mathbf{S}^{-1}\mathbf{L}^T\mathbf{D}^{-1}(\mathbf{I} - \mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\mathbf{U}^T)b \\ + \mathbf{S}^{-1}\mathbf{L}^T\mathbf{D}^{-1}(\mathbf{I} - \mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\mathbf{U}^T)\mathbf{D}\mathbf{U}\hat{\mathbf{L}}^{-T}\mathbf{V}^T\mathbf{H}^T\hat{\mathbf{F}}^{-1}f.\end{aligned}$$

Since $(\mathbf{I} - \mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\mathbf{U}^T)$ and $(\mathbf{I} - \mathbf{L}^T\mathbf{U}\hat{\mathbf{L}}^{-T}\mathbf{V}^T)$ are projection matrices, taking the norm of this error we obtain:

$$\begin{aligned}\|\delta x - \mathbf{V}\delta\hat{x}\| \leq \|\mathbf{S}^{-1}\|\|\mathbf{H}^T\hat{\mathbf{F}}^{-1}f\| \\ + \|\mathbf{S}^{-1}\mathbf{L}^T\mathbf{D}^{-1}\|\|b + \mathbf{D}\mathbf{U}\hat{\mathbf{L}}^{-T}\mathbf{V}^T\mathbf{H}^T\hat{\mathbf{F}}^{-1}f\|.\end{aligned} \tag{5.37}$$

Arbitrary projection can naturally lead to large errors, however we observe that taking $r = n$ results in square projection matrices, and a projection error of zero. By our requirement that $U^TV = I_r = I_n$, we have $U^T = V^{-1}$. Therefore $\hat{\mathbf{L}}^{-1} = \mathbf{U}^T\mathbf{L}^{-1}\mathbf{V}$ and hence the projection matrices $(\mathbf{I} - \mathbf{L}\mathbf{V}\hat{\mathbf{L}}^{-1}\mathbf{U}^T)$ and $(\mathbf{I} - \mathbf{L}^T\mathbf{U}\hat{\mathbf{L}}^{-T}\mathbf{V}^T)$ are equal to 0.

Furthermore, both parts of (5.37) contain the term $(d - \hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\mathbf{U}^T b)$ and thus we can hope to reduce the approximation error if $(d - \hat{\mathbf{H}}\hat{\mathbf{L}}^{-1}\mathbf{U}^T b)$ is small. Returning to our projected cost function (5.9):

$$\tilde{J}(\delta\hat{x}) = \frac{1}{2}\|\hat{\mathbf{L}}\delta\hat{x} - \hat{b}\|^2_{\hat{\mathbf{D}}^{-1}} + \frac{1}{2}\|d - \hat{\mathbf{H}}\delta\hat{x}\|^2_{\mathbf{R}^{-1}},$$

if the first part is solved exactly, we obtain $\hat{\mathbf{L}}^{-1}\mathbf{U}^T b = \delta\hat{x}$. Thus the term we wish to minimise in (5.37) becomes $(d - \hat{\mathbf{H}}\delta\hat{x})$. This is precisely the case if the second part of (5.9) is solved exactly. However it is not the case that both are solved exactly, nonetheless this presents a possible way to identify good projections.

Finding a sharp error bound for an arbitrary projection is generally not feasible. However when using the balanced truncation method to project the system, we can find an error bound.

For a stable linear time-invariant system (5.10) with an initial state of 0:

$$\begin{aligned} x_0 &= 0, \\ x_{k+1} &= \mathbf{A}x_k + \mathbf{B}u_k, \\ y_k &= \mathbf{C}x_k, \end{aligned}$$

the $\mathcal{H}_\infty$ norm of a linear time-invariant system (5.10) is given by

$$\|G\|_{\mathcal{H}_\infty} = \sup_{\omega \in \mathbb{R}} \sigma_{\max}(G(i\omega)),$$

where $\sigma_{\max}$ denotes the maximum singular value of a matrix, and

$$G(z) = (\mathbf{C}(zI - \mathbf{A})^{-1}\mathbf{B}),$$

is the *transfer function* of the system.

Applying the (discrete-time) Laplace transform (Z-transform) to the system (5.10), an explicit input-output relation can be obtained:

$$y(z) = (\mathbf{C}(zI - \mathbf{A})^{-1}\mathbf{B})u(z), \tag{5.38}$$

where $u(z)$ and $y(z)$ denote the Z-transforms of the input and output functions $u(t)$ and $y(t)$. Using Plancherel's theorem, the approximation error between the system (5.10) and the reduced system (5.11) is bounded by

$$\|y_k - \hat{y}_k\|_2 \leq \|G - \hat{G}\|_{\mathcal{H}_\infty} \|u_k\|_2,$$

with the respective transfer functions G and $\hat{G}$.

Thus for the discrete linear time-invariant system considered here:

$$\begin{aligned} \delta x_{-1} &= 0, \\ \delta x_{k+1} &= M\delta x_k + u_k, \\ d_k &= H\delta x_k, \end{aligned}$$

we observe the transfer function G corresponds to $G = (H(zI - M)^{-1})$. Hence when considering the data assimilation problem taking full observations with $H = I$, this allows us to consider an error bound on δx itself:

$$\|\delta x_k - V\delta\hat{x}_k\|_2 \leq \|G - \hat{G}\|_{\mathcal{H}_\infty} \|u_k\|_2,$$

where $G(z) = (zI - M)^{-1}$ and $\hat{G}(z) = V(zI - \hat{M})^{-1}$.

For balanced truncation, it is possible to derive a computable bound on the difference between the transfer functions of the full and reduced-order models. Taking the matrix of Hankel singular values Σ, we consider the truncated matrix of Hankel singular values which are kept in the reduced system (5.20),(5.21) Σ_r. We suppose the retained singular values are σ_i, with multiplicity m_i $i = 1, \ldots, k$, where $k < q$ and q is the total distinct Hankel singular values, then $\|G - \hat{G}\|_{\mathcal{H}_\infty}$ can be bounded by twice the sum of the distinct neglected Hankel singular values which are not retained:

$$\|G - \hat{G}\|_{\mathcal{H}_\infty} \leq 2(\sigma_{k+1} + \cdots \sigma_q). \tag{5.39}$$

For a proof we refer to [69].

5.6 | Numerical results

In this section we present numerical results for the projection methods introduced in Sections 5.3 and 5.4.

We present figures with the root mean squared error (RMSE) at each timestep of the assimilation window and forecast for each solution method. For these results the randomised methods were repeated 100 times and the mean RMSE is presented in the plots for comparison with the other methods.

We consider a coarse interpolatory projection as a simple method in order to compare to the other projections we consider. This is done considering the $n \times r$ matrix with a "1" in each column and the remainder to be "0" with the non-zero entries equally spaced. When taking $r = n$, this results in $V = U = I$.

The plots show the following methods: solving the full-sized system, projecting with coarse interpolatory projection matrices, generating the projection matrices through balanced truncation, and applying randomised projection matrices generated through a uniform distribution, a multivariate Gaussian distribution, and with a Rademacher distribution. These approaches are compared with the RMSE arising from evolving the background state of the system forward. We observe that the best results are obtained by using the full-sized system, and typically the worst results are those obtained from evolving the background state.

5.6.1 | One-dimensional advection-diffusion system

As a first example, we consider the one-dimensional (linear) advection-diffusion problem as in Chapter 3:

$$\frac{\partial}{\partial t}u(x,t) = 0.1\frac{\partial^2}{\partial x^2}u(x,t) + 1.4\frac{\partial}{\partial x}u(x,t) \tag{5.40}$$

for $x \in [0,1]$, $t \in (0,1)$, subject to the boundary and initial conditions

$$\begin{aligned} u(0,t) &= 0, && t \in (0,1) \\ u(1,t) &= 0, && t \in (0,1) \\ u(x,0) &= sin(\pi x), && x \in [0,1]. \end{aligned}$$

We discretise this system with a centered difference scheme for u_x and u_t, and a Crank-Nicolson scheme for u_{xx}, discretising x uniformly with $h = \frac{1}{500}$, and taking timesteps of size $\Delta t = 10^{-3}$.

We now consider this example as a data assimilation problem. We take an assimilation window of 200 timesteps, followed by a forecast of 800 timesteps.

The linear systems we must solve are of size $100,000$ for the full-sized problem, and $200r$ after applying a projection method. This is true for full or partial observations.

In the following figures, we consider the root mean squared error (RMSE) for the different approaches. The first 200 timesteps in the figure are the assimilation window, these are the timesteps where the observations of the system are taken, and these states are considered in the cost function being minimised. The subsequent timesteps are obtained by using our updated state to forecast forward.

Full observations

We first consider full, interpolatory observations ($p = 500$) in each timestep of the assimilation window, thus $H = I_{500}$, and we take the observation error covariance to be $R = 0.01I_{500}$. In this example we take $B_{i,j} = 0.1\exp(\frac{-|i-j|}{2n})$ as the background error covariance, with the model error covariance $Q = 10^{-6}I_{500}$.

Let us begin with a reduced space of size $r = 20$. The space the minimisation takes place in for the projected methods is thus 4% of the size of the full-size problem.

In Figure 5.1 a) we observe that the forecast resulting from solving the data

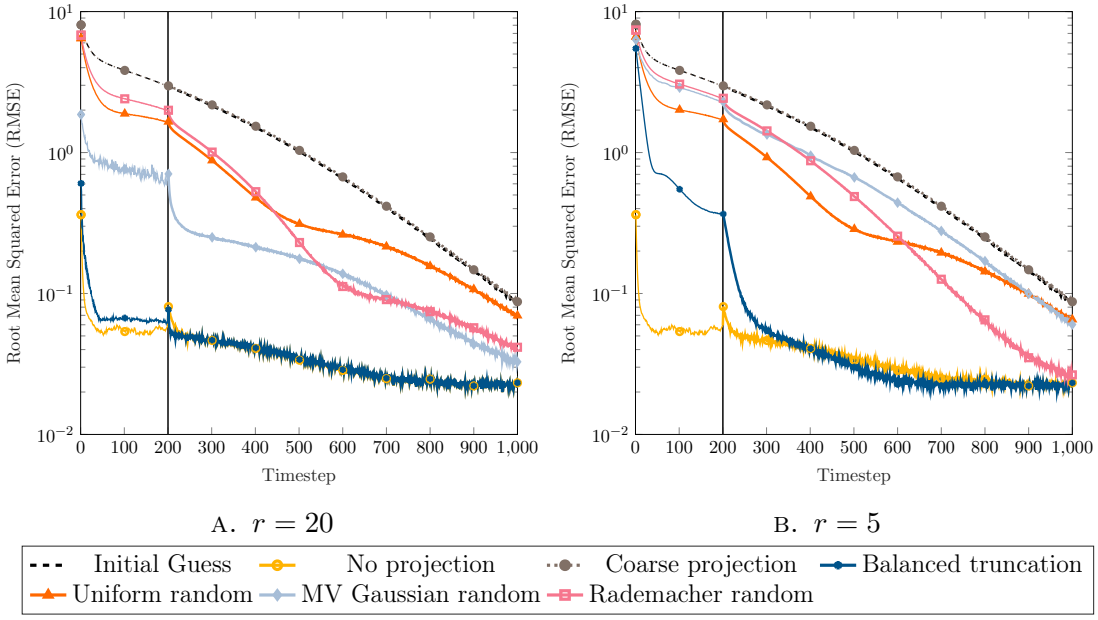


Figure 5.1: Root mean squared errors for the 1D advection-diffusion example with full, noisy observations ($r = 20$, $r = 5$).

assimilation problem using the coarse grid projection does not give any improvement over forecasting without performing assimilation. For the first 200 timesteps after the assimilation window, we observe the uniform random projection, and Rademacher random projection methods yield similarly accurate forecasts. After this point however the uniform random projection approach gives a less accurate forecast, though still more accurate than not performing data assimilation. In contrast, the Rademacher approach after 400 timesteps yields similar forecasts to the multivariate Gaussian projection method. The balanced truncation method achieves the best forecast out of the four projection methods, with the forecast resulting in the same level of error as the forecast obtained using the full-size model.

We consider in Figure 5.1 B) a smaller reduced space, here $r = 5$, which is just 1% of the size of the full-size models.

Despite this small space, the forecasts obtained using the balanced truncation method achieves results with the same level of error as the forecast obtained through the full-size model after 100 timesteps of the forecast window. During the assimilation window the method performs considerably less effectively than the example with $r = 20$ as the reduced system size.

The uniform random projection method perform very similarly to the $r = 20$ example, achieving comparable levels of error from the resulting forecast. The Rademacher random projection in this smaller space performs worse than the uniform random projection for the first 400 timesteps of the forecast window, after

which the resulting forecast has a smaller error.

In contrast the multivariate Gaussian approach results in significantly worse forecasts than in the larger space, though still better than not performing data assimilation.

Partial observations

We now consider partial observations in contrast to the full observations considered above, as in real-world applications, the number of observations is considerably less than the full state. Here we take $p = 100$ observations at each timestep of the assimilation window, with an observation in every fifth component, otherwise retaining the same setup as before, though here $R = 0.01I_{100}$. Let us again consider projecting with $r = 20$, and $r = 5$ where the resulting RMSE plots are shown in Figure 5.2.

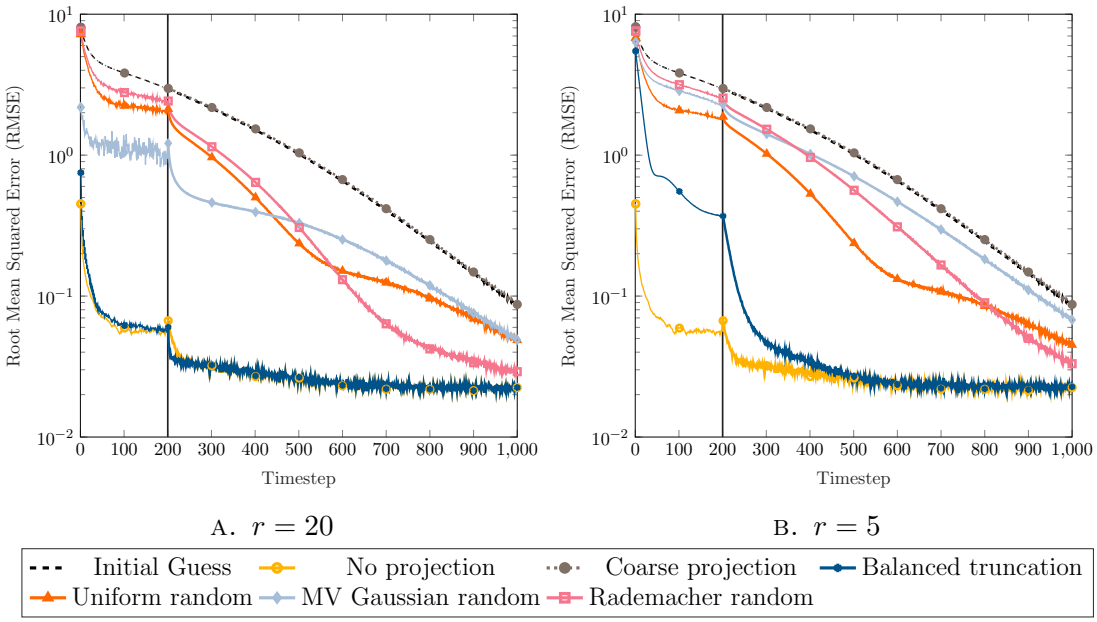


Figure 5.2: Root mean squared errors for the 1D advection-diffusion example with partial, noisy observations ($r = 20$, $r = 5$).

The resulting RMSEs for the data assimilation problem with partial observations in Figure 5.2 have similar characteristics to the full observations in Figure 5.1. The initial guess, and the forecasts generated using the coarse projection result in very similar RMSEs.

The forecasts obtained through the balanced truncation method, as in the full observation example achieves a similar error to the full-size model for the $r = 20$ case. In the $r = 5$ example, with partial observations it takes 300 timesteps of the

forecast window for the error from the forecast to be at the same level as the full-size model.

The multivariate Gaussian projection approach results in slightly higher RMSEs than in the full observation example, but in the $r = 20$ example still results in significantly lower error than not applying data assimilation.

For the Rademacher and uniform random projection methods, the resulting forecasts from the example with partial observations have less error in than the full observations example. Taking $r = 20$, both of these methods are more effective than the multivariate Gaussian approach after 300 timesteps of forecast. Whilst the Rademacher projection leads to the best results of the randomised projection methods after 400 timesteps of the forecast window.

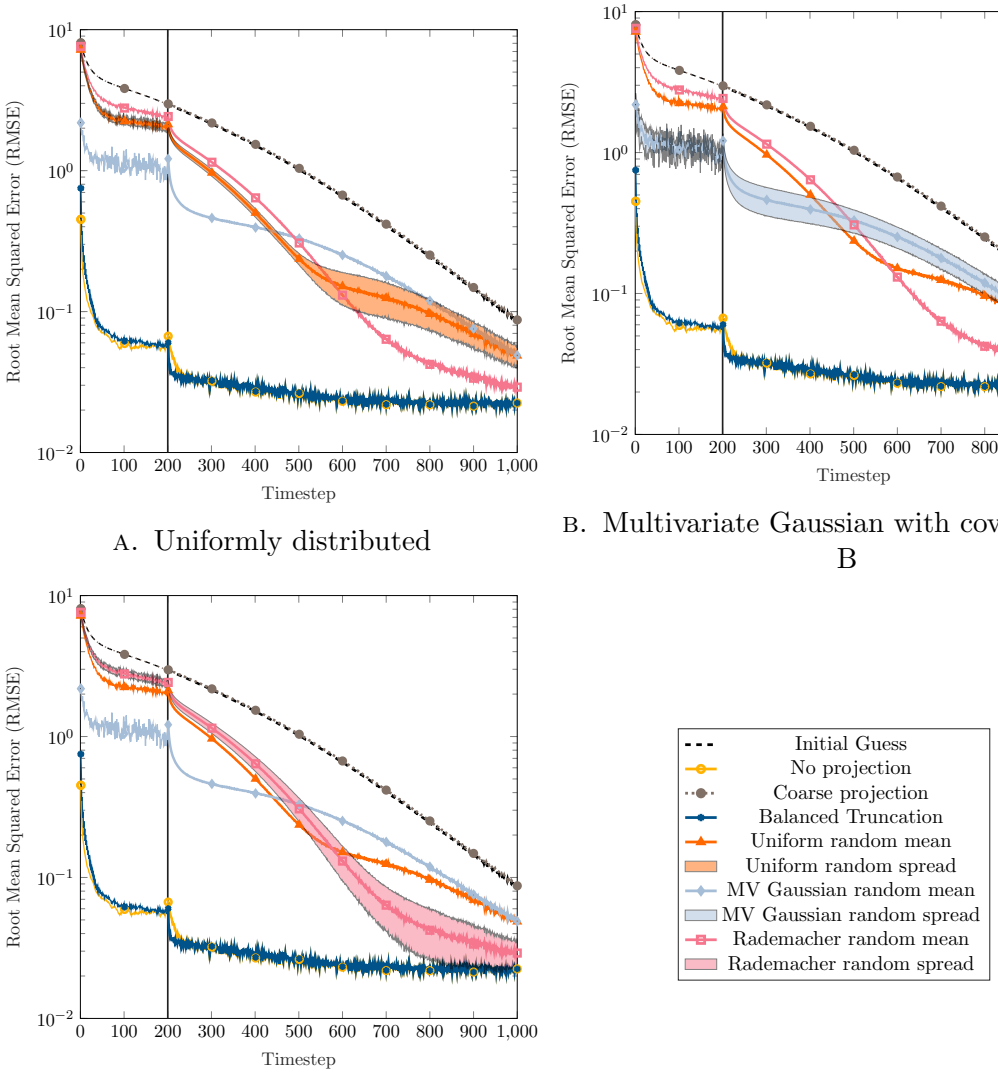
As with the full observation example, for the lower dimension case, $r = 5$, the uniform random projection remains similarly effective to the $r = 20$ example. However the Rademacher and multivariate Gaussian approaches again perform less successfully.

5.6.2 | The spread of randomised projection RMSEs

A consideration which has to be taken for the randomised methods which we present here is the variability of the methods depending on the random seed which has been taken. In the previous section, we presented the mean RMSE from 100 applications of the randomised projections for comparison to the other methods. In Figure 5.3 we consider the same example as in Figure 5.2 a), the advection-diffusion example with partial ($p = 100$), noisy observations taking covariance matrices $B_{i,j} = 0.1 \exp(\frac{-|i-j|}{2n})$ for the background, $Q = 10^{-6} I_{500}$ for the model, and $R = 0.01 I_{100}$ for the observation covariances. However, we present a shaded area of one standard deviation of the resulting forecasts above and below the mean.

In Figure 5.3, we observe that the forecasts obtained from the randomised projection methods are relatively similar to one another.

If we consider one standard deviation above and below the mean RMSEs obtained from these randomised projection methods we see that the uniform random projection forecasts we obtain have the same level of error as one another for the first 300 timesteps, at which point we see that this variability becomes slightly larger. The variability for the forecasts obtained using the Rademacher random projection increases over the forecast window, until the error level reaches that of the full-rank method after 700 timesteps of the forecast window. In contrast the multivariate



A. Uniformly distributed

B. Multivariate Gaussian with covariance B

C. Rademacher

Figure 5.3: Root mean squared errors for the 1D advection-diffusion example with partial, noisy observations ($r = 20$, $r = 5$) with spread of random methods.

Gaussian approach results in the variability of the forecasts decreasing over the course of the forecast window.

Computation time

In Table 5.1, we present a comparison of the computation times for different parts of the process in the advection-diffusion example using the projection methods presented in the numerical examples. We consider the advection-diffusion example used in Figure 5.2, taking partial observations ($p = 100$) for the advection-diffusion example with a state space of size 500 and 200 assimilation timesteps. Thus the

linear system we are solving is of size $100,000$ in the full-size problem, and $4,000$ or $1,000$ taking $r = 20$ and $r = 5$ respectively. We use Matlab's inbuilt conjugate gradient function to solve (5.34) and (5.35) for the full-size and projected problems respectively, applying the preconditioner $\mathbf{L}^T\mathbf{D}^{-1}\mathbf{L}$ or $\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\hat{\mathbf{L}}$. The stopping criteria for this method is a tolerance of 10^{-6} or 20 iterations. These computations were done on an Intel i5-4460 processor operating at 3.2GHz.

Projection method	Forming projection matrices	CG solve	Total
No proj.	0	5.0049	5.0049
BT ($r = 20$)	1.2271	0.1419	1.3690
Uniform ($r = 20$)	0.0284	0.0730	0.1014
MV Gaussian ($r = 20$)	0.0302	0.0760	0.1062
Rademacher ($r = 20$)	0.0287	0.0717	0.1004
Coarse proj. ($r = 20$)	0.0009	0.0208	0.0217
BT ($r = 5$)	1.1778	0.0467	1.2245
Uniform ($r = 5$)	0.0076	0.0257	0.0333
MV Gaussian ($r = 5$)	0.0092	0.0265	0.0357
Rademacher ($r = 5$)	0.0077	0.0257	0.0334
Coarse proj. ($r = 5$)	0.0007	0.0125	0.0132

Table 5.1: Comparison of computation time for different projection methods for the 1D advection-diffusion equation example ($r = 20$, $r = 5$).

We see in Table 5.1 that all the projection methods are significantly faster than performing the minimisation in the larger space. The balanced truncation method as mentioned previously requires considerable expense to compute the projection matrices U and V due to the requirement of solving Steins equation. As such when considering a smaller choice of r, the formation of the reduced matrices requires a similar amount of time.

5.6.3 | 2D linearised shallow water equations

As in Chapter 3 we take for a second example the two-dimensional linearised shallow water equations, with a constant phase velocity. We have two velocity components $u(x, y, t)$ and $v(x, y, t)$ and a height perturbation $\eta(x, y, t)$, where $(x, y) \in [0, 1]\times[0, 1]$ is a spacial coordinate and $t > 0$ is time. The governing PDEs are:

$$\frac{\partial u}{\partial t} = -\frac{\partial \eta}{\partial x}, \qquad \frac{\partial u}{\partial t} = -\frac{\partial \eta}{\partial y}, \qquad \frac{\partial \eta}{\partial t} = -\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)$$

with the initial conditions

$$u(x, y, 0) = 0, \qquad v(x, y, 0) = 0, \qquad \eta(x, y, 0) = \eta_0(x, y),$$

where $\eta_0(x, y)$ is a sinusoidal perturbation.

We solve this problem using centered finite differences, discretising the space with an $m \times m$ grid taking $m = 13$, thus leading to a state space size of $n = 507$ considering the height and two velocities, and taking timesteps of size $\Delta t = 5 \cdot 10^{-4}$.

As with the advection-diffusion example when considering this as a data assimilation problem, we take an assimilation window of $(N + 1) = 200$ timesteps, where observations are taken at each timestep, followed by a forecast of 800 timesteps. We compare the same projection methods as in Section 5.6.1, however as the spectrum of our model matrix M is not necessarily within the unit circle, we perform α-bounded balanced truncation (see Section 5.3) on the linear system. For this problem we take $\alpha \approx 1 + 7 \cdot 10^{-5}$, resulting in a stable $M_a = M/\alpha$.

For the following numerical examples, we consider the RMSE for just the height component of the state.

Full observations

As in the advection-diffusion example, let us first consider full, interpolatory observations here with $p = 507$, again giving the observation operator $H = I_p$. As before, we take the observation error covariance to be $R = 0.01 I_p$, the background error covariance $B_{i,j} = 0.1 \exp(\frac{-|i-j|}{2n})$, and the model error covariance $Q = 10^{-6} I_{507}$.

Let us begin with a reduced space of size $r = 20$, which as in the advection-diffusion example corresponds to 4% of the size of the full space. In Figure 5.4 a), we observe that the forecasts obtained by solving the data assimilation problem using the coarse interpolatory projection, the uniform random projection and the Rademacher random projection methods do not result in an improvement for the RMSE over not performing data assimilation.

As in the advection-diffusion example, the balanced truncation method achieves the best forecast for the projection methods, though here the RMSE is greater than using the full system. The multivariate Gaussian approach results in a similar forecast to balanced truncation, but as seen in Table 5.2, it is cheaper to compute.

Unfortunately applying a coarse projection or the two other randomised projections, taking a uniform or Rademacher projection do not result in better forecasts than without applying data assimilation.

For the smaller space with $r = 5$ which results in a space 1% of the size of the full-

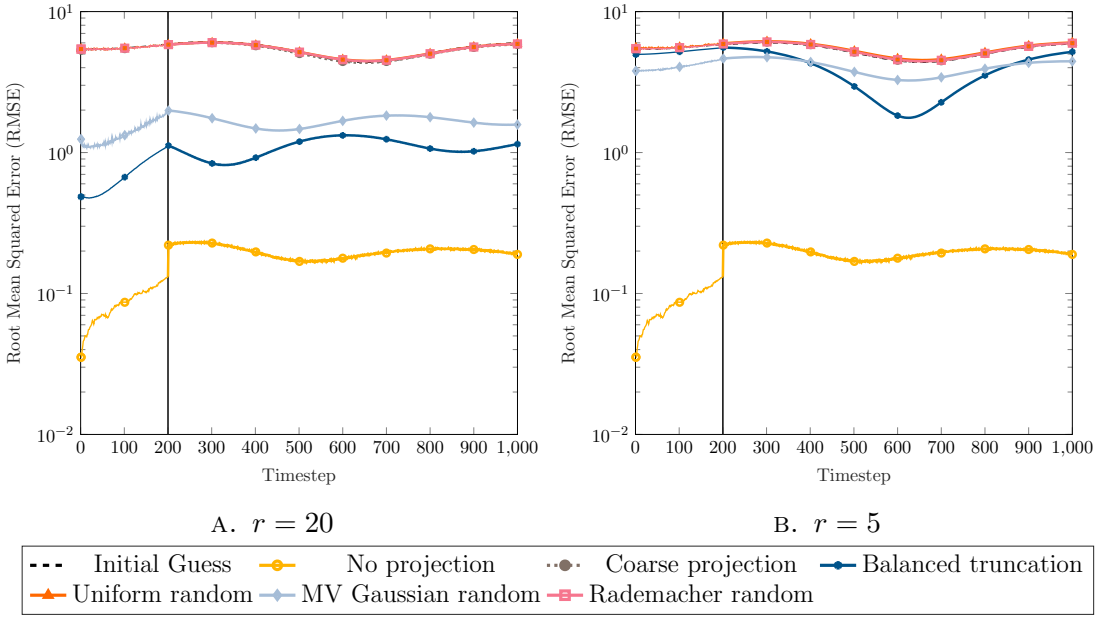


Figure 5.4: Root mean squared errors for the 2D shallow water equations example with full, noisy observations ($r = 20$, $r = 5$).

size problem in Figure 5.4 b), the projection methods are all less effective, with the coarse, uniform random, and Rademacher random projections all resulting in forecasts with the same levels of error as not applying data assimilation. In contrast, the multivariate Gaussian approach results in a forecast which is a slight improvement throughout the forecast window, whilst the balanced truncation method results in a forecast with a smaller error in the middle of the forecast window.

Partial observations

Considering partial observations, taking $p = 100$, and otherwise retaining the same setup as before, though here $R = 0.01I_{100}$. We obtain very similar results, presented in Figure 5.4 for $r = 20$ and $r = 5$.

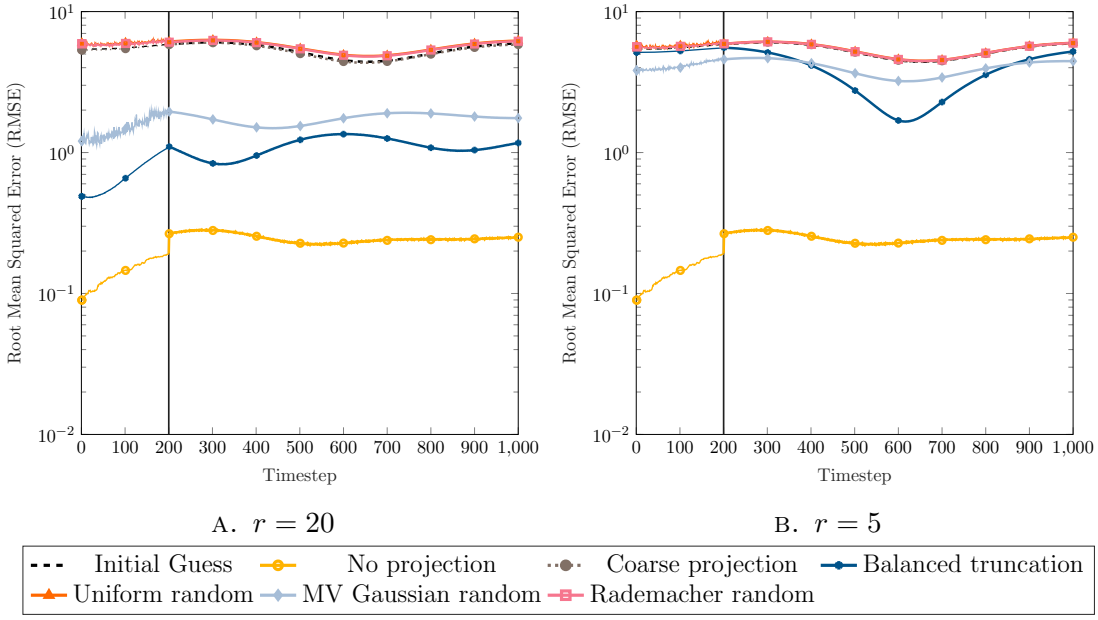


Figure 5.5: Root mean squared errors for the 2D shallow water equations example with partial, noisy observations ($r = 20$, $r = 5$).

Though not presented in this work, experiments with the choice of $r = 50$ displayed similar behaviour to the $r = 20$ examples for the shallow water equations, with both full and partial observations. The uniform and Rademacher randomised projection methods did not result in an improved forecast over those obtained from not applying data assimilation.

Computation time

In Table 5.2, we present a comparison of the computation times for different parts of the process in the two dimensional shallow water equations example using the projection methods presented in the numerical examples. We consider the example from Figure 5.5, with partial observations ($p = 100$) of the state of size 507, with 200 assimilation timesteps. The resulting linear systems are $101,400$ for the full-size problem and $10,000$ and $4,000$ taking $r = 20$ or $r = 5$. As with the advection-diffusion example, we use Matlab's inbuilt conjugate gradient function to solve (5.34) and (5.35) for the full-size and projected problems respectively, applying the preconditioner $\mathbf{L}^T\mathbf{D}^{-1}\mathbf{L}$ or $\hat{\mathbf{L}}^T\hat{\mathbf{D}}^{-1}\hat{\mathbf{L}}$. The stopping criteria for this method is a tolerance of 10^{-6} or 20 iterations. These computations were done on an Intel i5-4460 processor operating at 3.2GHz.

We see in Table 5.2 that the randomised and coarse projections results in a faster CG solve than the full-size system due to the smaller size, and forming the matrices

Projection method	Forming projection matrices	CG solve	Total
No proj.	0	0.1943	0.1943
BT ($r = 20$)	1.2887	0.1996	1.4884
Uniform ($r = 20$)	0.0288	0.1000	0.1288
MV Gaussian ($r = 20$)	0.0307	0.1047	0.1354
Rademacher ($r = 20$)	0.0291	0.1001	0.1292
Coarse proj. ($r = 20$)	0.0010	0.0189	0.0199
BT ($r = 5$)	1.2514	0.0512	1.3025
Uniform ($r = 5$)	0.0077	0.0286	0.0364
MV Gaussian ($r = 5$)	0.0093	0.0301	0.0394
Rademacher ($r = 5$)	0.0077	0.0286	0.0363
Coarse proj. ($r = 5$)	0.0007	0.0161	0.0168

Table 5.2: Comparison of computation time for different projection methods for the 2D shallow water equations example ($r = 20$, $r = 5$).

does not require much expense. In contrast the balanced truncation method as mentioned in Section 5.6.1, requires considerable expense to compute the projected matrices due to the Stein equation solves, furthermore we must compute a suitable α for the α-bounded balanced truncation adding an additional expense. Here we observe that the small number of iterations needed for the full-sized system CG solve allows it to perform effectively, and as such we do not see the same levels of savings in the CG solve as we did in the advection-diffusion example. This is due to the eigenvalues of M being tightly clustered, which is not necessarily the case for the projected $\hat{M}$. In further cycles of assimilation however, the projected matrix $\hat{M}$ could potentially be reused, thus amortising the cost of formation, particularly for the balanced truncation method.

5.6.4 | Lorenz-95 system

Let us now consider the Lorenz-95 system [94] which is a chaotic nonlinear example, which is often used to represent real world data assimilation problems such as weather forecasting, and the other example we introduced in Chapter 3. This is a generalisation of the three-dimensional Lorenz system [93] to n dimensions. The model is defined by a system of n nonlinear ordinary differential equations

$$\frac{\mathrm{d}z^i}{\mathrm{d}t} = -z^{i-2}z^{i-1} + z^{i-1}z^{i+1} - z^i + f, \tag{5.41}$$

where $z = [z^1, z^2, \ldots, z^n]^T$ is the state of the system, and f is a forcing term. Taking $f = 8$, the Lorenz system exhibits chaotic behaviour [57, 94]. For this example, we take $n = 500$.

We solve (5.41) using a 4th order Runge-Kutta method in order to obtain

$$z_{k+1} = \mathcal{M}_k(z_k), \quad \text{where } z_k = [z_k^1, z_k^2, \ldots, z_k^n]^T, \tag{5.42}$$

where $\mathcal{M}_k$ is the nonlinear model operator which evolves the state z_k to z_{k+1}. As before $\mathcal{H}_k$ denotes the observation operator for the state z_k. To formulate the data assimilation problem, we generate the tangent linear model, and observation operators M_k and H_k by linearising $\mathcal{M}_k$ and $\mathcal{H}_k$ about z_k.

As in Section 5.6.1, let us now consider this example as a data assimilation problem. We take an assimilation window of $(N + 1) = 200$ timesteps, where observations at each timestep, with a forecast of only 300 timesteps.

We compare the same projection methods as in Section 5.6.1, however as the Lorenz-95 system is not a stable, time-invariant system we cannot perform balanced truncation in the standard way. The generation of the projection matrices U^T and V is performed with the final linearised model matrix during the assimilation window, M_N. However, as the spectrum of this matrix is not within the unit circle, we perform α-bounded balanced truncation (see Section 5.3) on the linear system with M_N. From experimentation we observe similar results taking different choices of M to generate our projection matrices. For this problem we take $\alpha \approx 1.034$. This approach is not optimal however allows an illustrative comparison for balanced truncation to the randomised projection methods.

Full, noisy observations

As with our previous examples, let us consider full, interpolatory observations taking $p = 500$, which results in the observation operator $H = I_p$. As before, we take the observation error covariance to be $R = 0.01 I_p$, the background error covariance $B_{i,j} = 0.1 \exp(\frac{-|i-j|}{2n})$, and the model error covariance $Q = 10^{-6} I_{500}$.

In Figure 5.6 we observe that none of the projection methods see a significant improvement over not performing assimilation after the assimilation window, despite a better approximation within the assimilation window. Balanced truncation and the multivariate Gaussian projection method both achieve a significant improvement during the first half of the assimilation window. A possible reason for this is that the Lorenz system is a chaotic nonlinear system, and these nonlinearities may be too severe to be accurately captured in the projected model matrices. Further

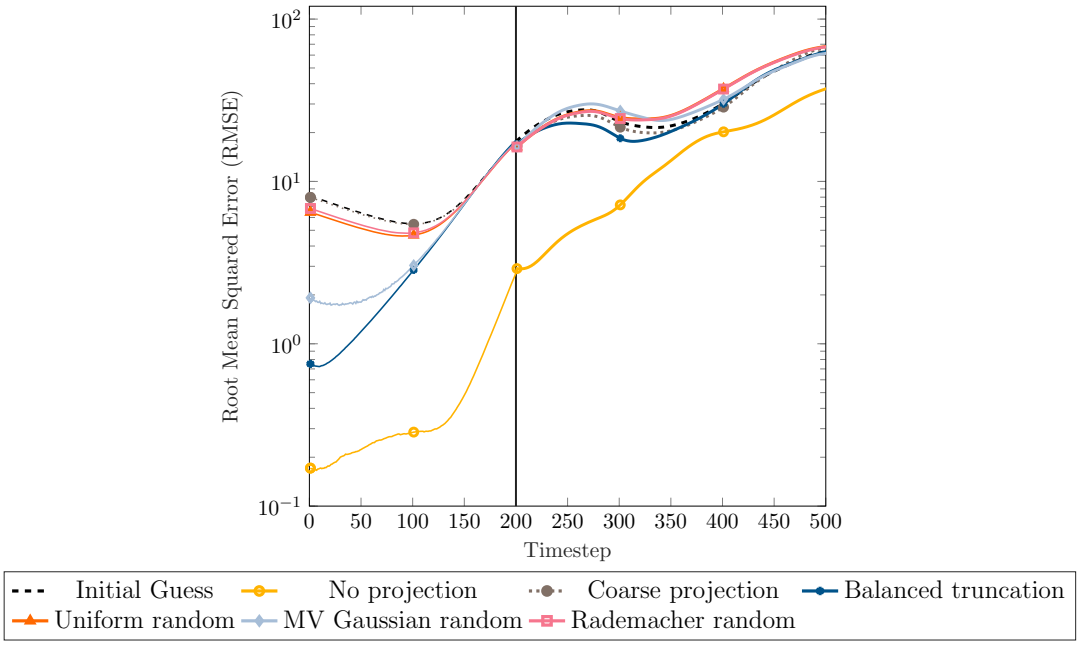


Figure 5.6: Root mean squared errors for the Lorenz-95 example with full, noisy observations ($r = 20$).

investigation into extensions of balanced truncation to nonlinear models, or utilising nonlinear model reduction methods such as POD may lead to improved performance for this problem. The improvement shown by the multivariate Gaussian projection method and balanced truncation over the first half of the assimilation window may suggest that if a shorter assimilation window were taken, more effective forecasts may be obtained in comparison to using the full model.

5.7 | Conclusions

The minimisation problem within weak constraint four-dimensional variational data assimilation usually needs to be solved in very large dimensions. In this chapter we have proposed projecting this problem to a space with a reduced dimension, which results in a reduction of computational expense. In particular we have applied projection methods to the problem, extending the use of balanced truncation to the weak constraint case. Furthermore we introduced randomised projection methods which are very cheap to implement, yet resulted in an effective method for some examples.

We considered the error resulting from these projection methods, and observe that in some scenarios we can obtain a small error for the projection, dependent on the system which we are considering. In the case of balanced truncation, which has

additional requirements on the model operator being stable, there exists a tractable error bound which can be applied here.

Numerical experiments have demonstrated that some randomised projections can compete with the balanced truncation method of model reduction within the data assimilation setting. Furthermore in these examples we achieved close levels of error to those obtained using the full scale minimisation, significantly better than not applying data assimilation despite the reduced space being significantly smaller.

The nonlinear and chaotic Lorenz-95 system does not respond well to the projection approaches investigated here. As such, further investigation is required into applying projection methods, both randomised and deterministic to these problems. Despite this, we have shown there are potential savings to be made by considering projections within weak constraint 4D-Var.

CHAPTER 6

CONCLUSION AND FURTHER WORK

Data assimilation is an important method for incorporating data (typically observations) into a model in order to create more accurate estimates of the actual state of the system. Performing data assimilation can typically be an expensive process with the models used in the data assimilation method often arising from physical processes and are often computationally expensive to evaluate themselves. A further property which majority of applications share is the vast dimensionality of the state vectors involved.

In this thesis we have introduced methods to reduce the size of the state space within the data assimilation process in order to reduce storage requirements and computation time. In particular we considered the weak constraint four dimensional variational data assimilation approach (weak constraint 4D-Var) and achieved this reduction in two different ways. In this final chapter we briefly summarise the findings from the previous chapters.

In Chapter 3 we considered the saddle point formulation of weak constraint four-dimensional variational data assimilation. We proposed a low-rank approach which approximates the solution to the saddle point system, with significant reductions in the storage needed. This was achieved by considering the structure of this saddle point system and using techniques from the theory of matrix equations. Using the properties of the Kronecker product we showed that low-rank solutions to the data assimilation problem exist under certain assumptions, with numerical experimentation demonstrating that this may be the case even when these assumptions are relaxed. We introduced a low-rank GMRES solver and considered the requirements for implementing this algorithm. Numerical experiments demonstrated that the low-rank approach introduced here is successful using both linear and nonlinear models.

In these examples we achieved close approximations to the full-rank solutions with storage requirements as low as 1% of those needed by the full-rank approach, which can be obtained in less time than through GMRES.

In Chapter 4 we presented three different preconditioners and applied them to the data assimilation saddle point problem, using both GMRES and the low-rank GMRES method introduced in Chapter 3. We considered the inexact constraint preconditioner [17, 18], and block diagonal and triangular Schur complement preconditioners. In order to apply these tractably, approximations for the $(2,1)$ block of the saddle point matrix were necessary.

We observed that when solving the data assimilation saddle point problem using either approach, the most effective preconditioner was the inexact constraint preconditioner approximating the model operator with the identity, and not including the observation operator. We also considered using the exact model operator and using properties of the Kronecker product to approximate the inverse, and whilst we did achieve superior results for close approximations to the inverse, it resulted in a larger number of matrix vector products and greater computational expense.

Preconditioning was not as effective in LR-GMRES when compared to GMRES. In our numerical examples, we observed that for the preconditioned and unpreconditioned system, the convergence of the residual stagnated. A possible explanation for this is that the low-rank approach acts like a regularisation, and hence in some sense like a projected preconditioner itself. In majority of examples using no preconditioner returned the smallest residual for the first 10 or more iterations. This may be due to the structure of the data assimilation saddle point matrix, with the more computationally expensive blocks being the $(1,2)/(2,1)$ blocks in contrast to the $(1,1)$ block as in other applications of saddle point matrices. As a result, further investigation into different types of preconditioners for this problem may be required.

Lastly in Chapter 5 we proposed projecting the minimisation problem within weak constraint 4D-Var to a space with a reduced dimension, which results in a reduction of computational expense. In particular we extended the use of balanced truncation to the weak constraint case, and introduced randomised projection methods which are very cheap to implement, yet resulted in an effective method for some examples.

We considered the error resulting from these projection methods, and observed that in some scenarios we can obtain a small error for the projection, dependent on the system which we are considering. In the case of balanced truncation, which has additional requirements on the model operator being stable, there exists a tractable

error bound which can be applied here.

Numerical experiments demonstrated that some randomised projections can compete with the balanced truncation method of model reduction within the data assimilation setting. Furthermore in these examples we achieved close levels of error to those obtained using the full scale minimisation, significantly better than not applying data assimilation despite the reduced space being significantly smaller. The nonlinear and chaotic Lorenz-95 system did not respond well to the projection approaches investigated here. As such, further investigation is required into applying projection methods, both randomised and deterministic to these problems. Despite this, we have shown there are potential savings to be made by considering projections within weak constraint 4D-Var.

Finally, we suggest some possible avenues for future research, building on, or extending some of the ideas raised in this thesis.

From Chapter 3, we consider the following topics.

- There has been recent development of iterative solvers designed specifically for saddle point problems, a family of saddle point minimum residual solvers (SPMR) introduced in [47]. The ideas used here for LR-GMRES could be extended to form low-rank SPMR methods, which may be more effective solvers for the data assimilation saddle point problem.

- As noted in Chapter 3, the LR-GMRES method leads to an inexact Krylov subspace method. Due to the truncation steps during the algorithm the matrix vector products are inexactly applied and thus this method does not satisfy standard GMRES and Krylov subspace properties. Analysis of the LR-GMRES method using inexact Krylov subspace literature [119] may lead to further understanding of the method.

Preconditioning the data assimilation problem as considered in Chapter 4 motivates a number of ideas for further research:

- The preconditioners considered here are used for saddle point problems across different applications. The data assimilation saddle point problem introduces an unusual situation where the $(1,2)$ block is more computationally expensive than the $(1,1)$ block. Further investigation into preconditioners for problems with this structure may result in better performance of iterative solvers.

- Due to the prevalence of problems where the $(1,1)$ block is more computationally expense, much analysis of Schur complement preconditioners makes the assumption that the exact $(1,2)$ block is used in the preconditioner. Analysis

of Schur complement preconditioners when using the exact $(1,1)$ block, but an approximation to the $(1,2)$ block may lead to further understanding of the convergence properties observed in Section 4.3 for these preconditioners.

- When considering the convergence of preconditioners for LR-GMRES in Section 4.5, we suggested that a "hybrid" approach, where no preconditioner is used for the first $10-20$ iterations before applying a preconditioner may yield better convergence, however was not considered in this work.

- The reduced model matrices generated in Chapter 5 provide an approximation of the matrices in the data assimilation saddle point problem. These could result in effective (and in the case of the randomised projection method, cheap) preconditioners for the data assimilation problem solved using GMRES.

Further topics which build on Chapter 5 include the following:

- For majority of applications of data assimilation, assimilations are performed in cycles, with the previous forecast informing the background estimate for the subsequent assimilation. Reusing the projection matrices constructed in Chapter 5 for a second cycle of assimilation may lead to improved performance for nonlinear problems.

- The model reduction method POD for nonlinear problems has previously been considered for the data assimilation problem [30]. A comparison between the projection methods considered here and projections obtained using a POD basis for linear and nonlinear examples would be interesting, in particular for a cycled data assimilation process as suggested above.

In addition we suggest some other investigations which could be undertaken:

- In this thesis new reduced rank approaches to the data assimilation problem have been proposed. In Chapter 2 we outlined a number of previous reduced rank methods for data assimilation. A comparison between the approaches proposed here and existing methods would be of interest. Furthermore, it is possible that the methods proposed in this thesis would work effectively in tandem with a reduced sequential method as a hybrid approach.

- There are many other model reduction techniques such as dynamic mode decomposition, reduced basis approaches, and IRKA (Iterative rational Krylov algorithm) which have not been considered in this thesis. These methods may yield more effective reduced models when applied to the data assimilation problem.

CHAPTER 7

BIBLIOGRAPHY

[1] D. Achlioptas, *Database-friendly random projections: Johnson-Lindenstrauss with binary coins*, J. Comput. Syst. Sci., 66 (2003), pp. 671–687. (Cited on page 112.)

[2] B. D. Anderson and J. B. Moore, *Optimal filtering*, Prentice-Hall, 1979. (Cited on page 10.)

[3] J. L. Anderson, *An ensemble adjustment Kalman filter for data assimilation*, Mon. Weather Rev., 129 (2001), pp. 2884–2903. (Cited on page 18.)

[4] A. C. Antoulas, *Approximation of large-scale dynamical systems*, vol. 6, SIAM, 2005. (Cited on pages 19, 22, 105, and 107.)

[5] M. Asch, M. Bocquet, and M. Nodet, *Data Assimilation: Methods, algorithms and applications*, SIAM, 2016. (Cited on pages 1, 14, 15, 17, and 21.)

[6] E. Atkins, M. Morzfeld, and A. J. Chorin, *Implicit particle methods and their connection with variational data assimilation*, Mon. Weather Rev., 141 (2013), pp. 1786–1803. (Cited on page 10.)

[7] R. N. Bannister, *A review of operational methods of variational and ensemble-variational data assimilation*, Q. J. R. Meteorol. Soc., 143 (2017), pp. 607–633. (Cited on page 1.)

[8] S. Barrachina, P. Benner, E. Quintana-Orti, and G. Quintana-Orti, *Parallel algorithms for balanced truncation of large-scale unstable systems*, in Proceedings of the 44th IEEE Conference on Decision and Control, IEEE, 2005. (Cited on page 108.)

[9] P. Benner and T. Breiten, *Low rank methods for a class of generalized Lyapunov equations and related issues*, Numer. Math., 124 (2013), pp. 441–470. (Cited on pages 33 and 38.)

[10] P. Benner and Z. Bujanović, *On the solution of large-scale algebraic Riccati equations by using low-dimensional invariant subspaces*, Linear Algebra Appl., 488 (2016), pp. 430–459. (Cited on pages 31 and 82.)

[11] P. Benner, G. E. Khoury, and M. Sadkane, *On the squared Smith method for large-scale Stein equations*, Numer. Linear Algebra Appl., 21 (2013), pp. 645–665. (Cited on page 105.)

[12] P. Benner, J.-R. Li, and T. Penzl, *Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems*, Numer. Linear Algebra Appl., 15 (2008), pp. 755–777. (Cited on page 25.)

[13] P. Benner, R.-C. Li, and N. Truhar, *On the ADI method for Sylvester equations*, J. Comput. Appl. Math., 233 (2009), pp. 1035–1045. (Cited on pages 31 and 82.)

[14] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer., 14 (2005), pp. 1–137. (Cited on pages 29, 58, 59, 63, 65, and 73.)

[15] M. Benzi and A. J. Wathen, *Some preconditioning techniques for saddle point problems*, Springer-Verlag, 2008, pp. 195–211. (Cited on pages 29, 59, and 63.)

[16] L. Bergamaschi, *On eigenvalue distribution of constraint-preconditioned symmetric saddle point matrices*, Numer. Linear Algebra Appl., 19 (2011), pp. 754–772. (Cited on pages 29, 66, and 97.)

[17] L. Bergamaschi, J. Gondzio, M. Venturin, and G. Zilli, *Inexact constraint preconditioners for linear systems arising in interior point methods*, Comput. Optim. Appl., 36 (2007), pp. 137–147. (Cited on pages 29, 66, 97, and 132.)

[18] ——, *Erratum to: Inexact constraint preconditioners for linear systems arising in interior point methods*, Comput. Optim. Appl., 49 (2009), pp. 401–406. (Cited on pages 29, 66, 97, and 132.)

[19] D. Bernstein, L. Davis, and D. Hyland, *The optimal projection equations for reduced-order, discrete-time modeling, estimation, and control*, J. Guid. Control Dyn., 9 (1986), pp. 288–293. (Cited on page 109.)

[20] E. Bingham and H. Mannila, *Random projection in dimensionality reduction: Applications to image and text data*, in Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '01, New York, NY, USA, 2001, ACM, pp. 245–250. (Cited on pages 100, 111, and 112.)

[21] C. H. Bishop, B. J. Etherton, and S. J. Majumdar, *Adaptive sampling with the ensemble transform Kalman filter. part I: Theoretical aspects*, Mon. Weather Rev., 129 (2001), pp. 420–436. (Cited on page 18.)

[22] É. Blayo, M. Bocquet, E. Cosme, and L. F. Cugliandolo, eds., *Advanced Data Assimilation for Geosciences*, Oxford University Press, 2014. (Cited on page 1.)

[23] C. Boess, *Using model reduction techniques within the Incremental 4D-Var method*, PhD thesis, University of Bremen, 2008. (Cited on pages 20, 23, 99, 100, 103, 108, 109, and 111.)

[24] C. Boess, A. Lawless, N. Nichols, and A. Bunse-Gerstner, *State estimation using model order reduction for unstable systems*, Comput. Fluids, 46 (2011), pp. 155–160. (Cited on pages 20, 23, 99, 100, 103, and 109.)

[25] B. Bonan, M. Nodet, O. Ozenda, and C. Ritz, *Data assimilation in glaciology*, in Advanced Data Assimilation for Geosciences, Oxford University Press, 2014, pp. 577–584. (Cited on page 1.)

[26] P. Brasseur and J. Verron, *The SEEK filter method for data assimilation in oceanography: a synthesis*, Ocean Dyn., 56 (2006), pp. 650–661. (Cited on page 15.)

[27] K. L. Brown, I. Gejadze, and A. Ramage, *A multilevel approach for computing the limited-memory Hessian and its inverse in variational data assimilation*, SIAM J. Sci. Comput., 38 (2016), pp. A2934–A2963. (Cited on page 21.)

[28] M. Buehner, P. L. Houtekamer, C. Charette, H. L. Mitchell, and B. He, *Intercomparison of variational data assimilation and the ensemble Kalman filter for global deterministic NWP. part I: Description and*

single-observation experiments, Mon. Weather Rev., 138 (2010), pp. 1550–1566. (Cited on pages 10 and 18.)

[29] ——, *Intercomparison of variational data assimilation and the ensemble Kalman filter for global deterministic NWP. part II: One-month experiments with real observations*, Mon. Weather Rev., 138 (2010), pp. 1567–1586. (Cited on page 10.)

[30] Y. Cao, J. Zhu, I. M. Navon, and Z. Luo, *A reduced-order approach to four-dimensional variational data assimilation using proper orthogonal decomposition*, Internat. J. Numer. Methods Fluids, 53 (2007), pp. 1571–1583. (Cited on pages 22 and 134.)

[31] S. Chaturantabut and D. C. Sorensen, *Nonlinear model reduction via discrete empirical interpolation*, SIAM J. Sci. Comput., 32 (2010), pp. 2737–2764. (Cited on pages 13 and 23.)

[32] S. E. Cohn, N. S. Sivakumaran, and R. Todling, *A fixed-lag Kalman smoother for retrospective data assimilation*, Mon. Weather Rev., 122 (1994), pp. 2838–2867. (Cited on page 10.)

[33] S. E. Cohn and R. Todling, *Approximate data assimilation schemes for stable and unstable dynamics*, J. Meteor. Soc. Japan Ser. II, 74 (1996), pp. 63–75. (Cited on page 14.)

[34] P. Courtier, J. Derber, R. Errico, J.-F. Louis, and T. Vukićević, *Important literature on the use of adjoint, variational methods and the Kalman filter in meteorology*, Tellus A, 45 (1993), pp. 342–357. (Cited on page 20.)

[35] P. Courtier, J.-N. Thépaut, and A. Hollingsworth, *A strategy for operational implementation of 4D-Var, using an incremental approach*, Q. J. R. Meteorol. Soc., 120 (1994), pp. 1367–1387. (Cited on pages 9, 26, and 100.)

[36] J. Crank and P. Nicolson, *A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type*, in Math. Proc. Cambridge Philos. Soc., vol. 43, Cambridge Univ Press, 1947, pp. 50–67. (Cited on page 42.)

[37] D. N. Daescu and I. M. Navon, *Efficiency of a POD-based reduced second-order adjoint model in 4D-Var data assimilation*, Internat. J. Numer. Methods Fluids, 53 (2007), pp. 985–1004. (Cited on page 22.)

[38] S. Dasgupta and A. Gupta, *An elementary proof of a theorem of Johnson and Lindenstrauss*, Random Structures Algorithms, 22 (2002), pp. 60–65. (Cited on page 112.)

[39] E. de Sturler and J. Liesen, *Block-diagonal and constraint preconditioners for nonsymmetric indefinite linear systems. part I: Theory*, SIAM J. Sci. Comput., 26 (2005), pp. 1598–1619. (Cited on page 64.)

[40] L. Debreu, E. Neveu, E. Simon, F.-X. L. Dimet, and A. Vidard, *Multigrid solvers and multigrid preconditioners for the solution of variational data assimilation problems*, Q. J. R. Meteorol. Soc., 142 (2015), pp. 515–528. (Cited on page 21.)

[41] G. Desroziers, J.-T. Camino, and L. Berre, *4DEnVar: link with 4D state formulation of variational assimilation and different possible implementations*, Q. J. R. Meteorol. Soc., 140 (2014), pp. 2097–2110. (Cited on pages 10 and 18.)

[42] F.-X. L. Dimet and O. Talagrand, *Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects*, Tellus A, 38 (1986), pp. 97–110. (Cited on page 7.)

[43] P. Drineas and M. W. Mahoney, *RandNLA*, Commun. ACM, 59 (2016), pp. 80–90. (Cited on page 100.)

[44] V. Druskin, L. Knizhnerman, and V. Simoncini, *Analysis of the rational Krylov subspace and ADI methods for solving the Lyapunov equation*, SIAM J. Numer. Anal., 49 (2011), pp. 1875–1898. (Cited on page 26.)

[45] S. Durbiano, *Vecteurs caractéristiques de modèles océaniques pour la réduction d'ordre en assimilation de données*, PhD thesis, Université Joseph Fourier Grenoble, 2001. (Cited on pages 21 and 22.)

[46] D. Enns, *Model reduction for control system design*, PhD thesis, Department of Aeronautics and Astronautics, Stanford University, 1984. (Cited on page 108.)

[47] R. Estrin and C. Greif, *SPMR: A family of saddle-point minimum residual solvers*, SIAM J. Sci. Comput., 40 (2018), pp. A1884–A1914. (Cited on page 133.)

[48] G. Evensen, *Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics*, J. Geophys. Res., 99 (1994), pp. 10143–10162. (Cited on pages 14, 16, and 17.)

[49] B. F. Farrell and P. J. Ioannou, *State estimation using a reduced-order Kalman filter*, J. Atmos. Sci., 58 (2001), pp. 3666–3680. (Cited on pages 19 and 20.)

[50] M. Fisher, S. Gratton, S. Gürol, Y. Trémolet, and X. Vasseur, *Low rank updates in preconditioning the saddle point systems arising from data assimilation problems*, Optim. Method. Softw., 0 (2016), pp. 1–25. (Cited on page 66.)

[51] M. Fisher and S. Gürol, *Parallelisation in the time dimension of four-dimensional variational data assimilation*, Q. J. R. Meteorol. Soc., (2017). (Cited on page 8.)

[52] M. Fisher, M. Leutbecher, and G. A. Kelly, *On the equivalence between Kalman smoothing and weak-constraint four-dimensional variational data assimilation*, Q. J. R. Meteorol. Soc., 131 (2005), pp. 3235–3246. (Cited on pages 8 and 10.)

[53] M. Fisher, Y. Trémolet, H. Auvinen, D. Tan, and P. Poli, *Weak-constraint and long-window 4D-Var*, Tech. Rep. 655, ECMWF, 2011. (Cited on pages 7, 8, 18, 29, 66, and 67.)

[54] G. M. Flagg and S. Gugercin, *On the ADI method for the Sylvester equation and the optimal-$\mathcal{H}_2$ points*, Appl. Numer. Math., 64 (2013), pp. 50–58. (Cited on page 31.)

[55] M. A. Freitag and D. L. H. Green, *A low-rank approach to the solution of weak constraint variational data assimilation problems*, J. Comput. Phys., 357 (2018), pp. 263–281. (Cited on pages 7, 8, 14, 25, and 57.)

[56] ——, *Projection methods for weak constraint variational data assimilation*, submitted, (2019). (Cited on page 99.)

[57] M. A. Freitag and R. Potthast, *Synergy of inverse problems and data assimilation techniques*, vol. 13, Walter de Gruyter, 2013, pp. 1–53. (Cited on pages 6, 10, 52, and 128.)

[58] A. Gelb, ed., *Applied optimal estimation*, MIT Press, 1974. (Cited on pages 4 and 5.)

[59] M. Ghil, *Meteorological data assimilation for oceanographers. part I: Description and theoretical framework*, Dynam. Atmos. Oceans, 13 (1989), pp. 171–218. (Cited on page 1.)

[60] G. H. Golub and C. F. Van Loan, *Matrix computations*, vol. 3, Johns Hopkins University Press, 2012. (Cited on page 36.)

[61] L. Grasedyck, *Existence and computation of low Kronecker-rank approximations for large linear systems of tensor product structure*, Computing, 72 (2004), pp. 247–265. (Cited on pages 33 and 34.)

[62] ——, *Existence of a low rank or $\mathcal{H}$-matrix approximant to the solution of a Sylvester equation*, Numer. Linear Algebra Appl., 11 (2004), pp. 371–389. (Cited on pages 26 and 31.)

[63] S. Gratton, S. Gürol, E. Simon, and P. L. Toint, *Guaranteeing the convergence of the saddle formulation for weakly constrained 4D-Var data assimilation*, Q. J. R. Meteorol. Soc., 144 (2018), pp. 2592–2602. (Cited on page 7.)

[64] S. Gratton, A. S. Lawless, and N. K. Nichols, *Approximate Gauss–Newton methods for nonlinear least squares problems*, SIAM J. Optim., 18 (2007), pp. 106–132. (Cited on page 26.)

[65] A. Greenbaum, *Iterative methods for solving linear systems*, vol. 17, SIAM, 1997. (Cited on page 58.)

[66] N. Gustafsson, T. Janjić, C. Schraff, D. Leuenberger, M. Weissmann, H. Reich, P. Brousseau, T. Montmerle, E. Wattrelot, A. Bučánek, M. Mile, R. Hamdi, M. Lindskog, J. Barkmeijer, M. Dahlbom, B. Macpherson, S. Ballard, G. Inverarity, J. Carley, C. Alexander, D. Dowell, S. Liu, Y. Ikuta, and T. Fujita, *Survey of data assimilation methods for convective-scale numerical weather prediction at operational centres*, Q. J. R. Meteorol. Soc., 144 (2018), pp. 1218–1256. (Cited on page 1.)

[67] N. Halko, P. G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Rev., 53 (2011), pp. 217–288. (Cited on page 100.)

[68] T. M. Hamill and C. Snyder, *A hybrid ensemble Kalman filter – 3D variational analysis scheme*, Mon. Weather Rev., 128 (2000), pp. 2905–2919. (Cited on page 18.)

[69] D. Hinrichsen and A. Pritchard, *An improved error estimate for reduced-order models of discrete-time systems*, IEEE Trans. Autom. Control, 35 (1990), pp. 317–320. (Cited on page 117.)

[70] R. A. Horn and C. R. Johnson, *Matrix analysis*, Cambridge University Press, 2012. (Cited on page 36.)

[71] K. Ide, P. Courtier, M. Ghil, and A. C. Lorenc, *Unified notation for data assimilation: operational, sequential and variational*, J. Meteor. Soc. Japan, 75 (1997), pp. 181–189. (Cited on pages 20 and 23.)

[72] A. Jazwinski, *Stochastic processes and filtering theory*, Academic Press, 1970. (Cited on page 5.)

[73] K. Jbilou, *Low rank approximate solutions to large Sylvester matrix equations*, Appl. Math. Comput., 177 (2006), pp. 365–376. (Cited on page 105.)

[74] W. B. Johnson and J. Lindenstrauss, *Extensions of Lipschitz mappings into a Hilbert space*, Contemp. Math., 26 (1984), p. 1. (Cited on page 111.)

[75] R. E. Kalman, *A new approach to linear filtering and prediction problems*, J. Basic Eng., 82 (1960), pp. 35–45. (Cited on page 4.)

[76] S. Kim, N. Shepherd, and S. Chib, *Stochastic volatility: Likelihood inference and comparison with ARCH models*, Rev. Econ. Stud., 65 (1998), pp. 361–393. (Cited on page 1.)

[77] D. Kressner and C. Tobler, *Krylov subspace methods for linear systems with tensor product structure*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 1688–1714. (Cited on pages 25 and 33.)

[78] M. Krysta, E. Blayo, E. Cosme, and J. Verron, *A consistent hybrid variational-smoothing data assimilation method: Application to a simple shallow-water model of the turbulent midlatitude ocean*, Mon. Weather Rev., 139 (2011), pp. 3333–3347. (Cited on page 22.)

[79] S. Lall and C. Beck, *Error-bounds for balanced model-reduction of linear time-varying systems*, IEEE Trans. Automat. Control, 48 (2003), pp. 946–956. (Cited on page 108.)

[80] S. Lall, J. E. Marsden, and S. Glavaški, *A subspace approach to balanced truncation for model reduction of nonlinear control systems*, Internat. J. Robust Nonlinear Control, 12 (2002), pp. 519–535. (Cited on pages 14 and 108.)

[81] K. Law, A. Stuart, and K. Zygalakis, *Data Assimilation*, Springer International Publishing, 2015. (Cited on pages 6 and 10.)

[82] A. S. Lawless, *Variational data assimilation for very large environmental problems*, vol. 13, Walter de Gruyter, 2013, pp. 55–90. (Cited on pages 2 and 25.)

[83] A. S. Lawless, S. Gratton, and N. K. Nichols, *Approximate iterative methods for variational data assimilation*, Internat. J. Numer. Methods Fluids, 47 (2005), pp. 1129–1135. (Cited on page 26.)

[84] A. S. Lawless, N. K. Nichols, C. Boess, and A. Bunse-Gerstner, *Approximate Gauss–Newton methods for optimal state estimation using reduced-order models*, Internat. J. Numer. Methods Fluids, 56 (2008), pp. 1367–1373. (Cited on pages 20, 23, 99, 100, 103, and 109.)

[85] ——, *Using model reduction methods within incremental four-dimensional variational data assimilation*, Mon. Weather Rev., 136 (2008), pp. 1511–1522. (Cited on pages 20, 23, 99, 100, 103, and 109.)

[86] J.-R. Li and J. White, *Low-rank solution of Lyapunov equations*, SIAM J. Matrix Anal. Appl., 24 (2002), pp. 260–280. (Cited on page 26.)

[87] T. Li, P. C.-Y. Weng, E. K. wah Chu, and W.-W. Lin, *Large-scale Stein and Lyapunov equations, Smith method, and applications*, Numer. Algorithms, 63 (2012), pp. 727–752. (Cited on page 105.)

[88] J. Liesen and P. Tichý, *Convergence analysis of Krylov subspace methods*, GAMM-Mitteilungen, 27 (2004), pp. 153–173. (Cited on pages 29, 57, and 58.)

[89] Y. Lin, E. B. Le, D. O'Malley, V. V. Vesselinov, and T. Bui-Thanh, *Large-scale inverse model analyses employing fast randomized data reduction*, Water Resour. Res., 53 (2017), pp. 6784–6801. (Cited on pages 100, 111, and 112.)

[90] A. C. Lorenc, *Modelling of error covariances by 4D-Var data assimilation*, Q. J. R. Meteorol. Soc., 129 (2003), pp. 3167–3182. (Cited on pages 7 and 18.)

[91] A. C. Lorenc, N. E. Bowler, A. M. Clayton, S. R. Pring, and D. Fairbairn, *Comparison of hybrid-4DEnVar and hybrid-4DVar data assimilation methods for global NWP*, Mon. Weather Rev., 143 (2015), pp. 212–229. (Cited on page 18.)

[92] A. C. Lorenc and M. Jardak, *A comparison of hybrid variational data assimilation methods for global NWP*, Q. J. R. Meteorol. Soc., 144 (2018), pp. 2748–2760. (Cited on page 18.)

[93] E. N. Lorenz, *Deterministic nonperiodic flow*, J. Atmos. Sci., 20 (1963), pp. 130–141. (Cited on pages 51 and 127.)

[94] ——, *Predictability: A problem partly solved*, in Proc. Seminar on predictability, vol. 1, 1996. (Cited on pages 51, 52, 127, and 128.)

[95] M. W. Mahoney, *Randomized algorithms for matrices and data*, Found. Trends Mach. Learn., 3 (2010), pp. 123–224. (Cited on pages 100 and 112.)

[96] B. C. Moore, *Principal component analysis in linear systems: Controllability, observability, and model reduction*, IEEE Trans. Automat. Control, 26 (1981), pp. 17–32. (Cited on pages 19, 20, 23, 99, and 103.)

[97] M. F. Murphy, G. H. Golub, and A. J. Wathen, *A note on preconditioning for indefinite linear systems*, SIAM J. Sci. Comput., 21 (2000), pp. 1969–1972. (Cited on pages 64 and 73.)

[98] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975), pp. 617–629. (Cited on pages 29 and 57.)

[99] T. Penzl, *A cyclic low-rank Smith method for large sparse Lyapunov equations*, SIAM J. Sci. Comput., 21 (1999), pp. 1401–1418. (Cited on pages 25 and 26.)

[100] D. T. Pham, J. Verron, and M. C. Roubaud, *A singular evolutive extended Kalman filter for data assimilation in oceanography*, J. Marine Syst., 16 (1998), pp. 323–340. (Cited on pages 1, 14, and 15.)

[101] F. Rabier and M. Fisher, *Data assimilation in meteorology*, Oxford University Press, 2015, ch. 19, pp. 433–457. (Cited on page 1.)

[102] F. Rabier, H. Järvinen, E. Klinker, J.-F. Mahfouf, and A. Simmons, *The ECMWF operational implementation of four-dimensional variational assimilation. I: Experimental results with simplified physics*, Q. J. R. Meteorol. Soc., 126 (2000), pp. 1143–1170. (Cited on page 1.)

[103] G. Raskutti and M. W. Mahoney, *A statistical perspective on randomized sketching for ordinary least-squares*, J. Mach. Learn. Res., 17 (2016), pp. 1–31. (Cited on page 100.)

[104] F. Rawlins, S. P. Ballard, K. J. Bovis, A. M. Clayton, D. Li, G. W. Inverarity, A. C. Lorenc, and T. J. Payne, *The Met Office global four-dimensional variational data assimilation scheme*, Q. J. R. Meteorol. Soc., 133 (2007), pp. 347–362. (Cited on page 1.)

[105] C. Robert, E. Blayo, and J. Verron, *Comparison of reduced-order, sequential and variational data assimilation methods in the tropical Pacific Ocean*, Ocean Dyn., 56 (2006), pp. 624–633. (Cited on page 22.)

[106] ——, *Reduced-order 4D-Var: A preconditioner for the incremental 4D-Var data assimilation method*, Geophys. Res. Lett., 33 (2006). (Cited on pages 15 and 22.)

[107] C. Robert, S. Durbiano, E. Blayo, J. Verron, J. Blum, and F.-X. L. Dimet, *A reduced-order strategy for 4D-Var data assimilation*, J. Marine Syst., 57 (2005), pp. 70–82. (Cited on page 21.)

[108] M. Rozložník, *Saddle-point problems and their iterative solution*, Springer International Publishing, 2018. (Cited on pages 59 and 63.)

[109] T. Rusten and R. Winther, *A preconditioned iterative method for saddle-point problems*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 887–904. (Cited on page 61.)

[110] Y. Saad, *Numerical solution of large Lyapunov equations*, in Signal Processing, Scattering and Operator Theory, and Numerical Methods, Proc. MTNS-89, Birkhauser, 1990, pp. 503–511. (Cited on pages 25 and 26.)

[111] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Comput., 7 (1986), pp. 856–869. (Cited on pages 29, 38, and 57.)

[112] M. Sadkane, *A low-rank Krylov squared Smith method for large-scale discrete-time Lyapunov equations*, Linear Algebra Appl., 436 (2012), pp. 2807–2827. (Cited on page 105.)

[113] H. Sandberg and A. Rantzer, *Balanced truncation of linear time-varying systems*, IEEE Trans. Automat. Control, 49 (2004), pp. 217–229. (Cited on pages 14 and 108.)

[114] Y. Sasaki, *An objective analysis based on the variational method*, J. Meteor. Soc. Japan, 36 (1958), pp. 77–88. (Cited on page 6.)

[115] ——, *Some basic formalisms in numerical variational analysis*, Mon. Weather Rev., 98 (1970), pp. 875–883. (Cited on pages 6 and 7.)

[116] J. Scherpen, *Balancing for nonlinear systems*, Systems Control Lett., 21 (1993), pp. 143–153. (Cited on pages 14 and 108.)

[117] V. Simoncini, *A new iterative method for solving large-scale Lyapunov matrix equations*, SIAM J. Sci. Comput., 29 (2007), pp. 1268–1288. (Cited on pages 25, 26, and 82.)

[118] ——, *Computational methods for linear matrix equations*, SIAM Rev., 58 (2016), pp. 377–441. (Cited on pages 25, 31, and 105.)

[119] V. Simoncini and D. B. Szyld, *Theory of inexact Krylov subspace methods and applications to scientific computing*, SIAM J. Sci. Comput., 25 (2003), pp. 454–477. (Cited on pages 41 and 133.)

[120] P. J. Smith, S. L. Dance, M. J. Baines, N. K. Nichols, and T. R. Scott, *Variational data assimilation for parameter estimation: application to a simple morphodynamic model*, Ocean Dyn., 59 (2009), pp. 697–708. (Cited on page 1.)

[121] R. A. Smith, *Matrix equation $XA + BX = C$*, SIAM J. Appl. Math., 16 (1968), pp. 198–201. (Cited on page 105.)

[122] R. Ştefănescu, A. Sandu, and I. M. Navon, *Comparison of POD reduced order strategies for the nonlinear 2D shallow water equations*, Internat. J. Numer. Methods Fluids, 76 (2014), pp. 497–521. (Cited on page 23.)

[123] ——, *POD/DEIM reduced-order strategies for efficient four dimensional variational data assimilation*, J. Comput. Phys., 295 (2015), pp. 569–595. (Cited on pages 22 and 23.)

[124] F. Stenger, *Numerical methods based on Sinc and analytic functions*, Springer New York, 1993. (Cited on page 33.)

[125] M. Stoll and T. Breiten, *A low-rank in time approach to PDE-constrained optimization*, SIAM J. Sci. Comput., 37 (2015), pp. B1–B29. (Cited on pages 25, 29, 30, 38, 39, and 82.)

[126] O. Talagrand and P. Courtier, *Variational assimilation of meteorological observations with the adjoint vorticity equation. I: Theory*, Q. J. R. Meteorol. Soc., 113 (1987), pp. 1311–1328. (Cited on page 7.)

[127] M. Tanguay, P. Bartello, and P. Gauthier, *Four-dimensional data assimilation with a wide range of scales*, Tellus A, 47 (1995), pp. 974–997. (Cited on page 20.)

[128] X. Tian, Z. Xie, and A. Dai, *An ensemble-based explicit four-dimensional variational assimilation method*, J. Geophys. Res., 113 (2008). (Cited on page 22.)

[129] M. K. Tippett, J. L. Anderson, C. H. Bishop, T. M. Hamill, and J. S. Whitaker, *Ensemble square root filters*, Mon. Weather Rev., 131 (2003), pp. 1485–1490. (Cited on pages 14, 16, and 18.)

[130] Y. Trémolet, *Diagnostics of linear and incremental approximations in 4D-Var*, Q. J. R. Meteorol. Soc., 130 (2004), pp. 2233–2251. (Cited on page 20.)

[131] ——, *Accounting for an imperfect model in 4D-Var*, Q. J. R. Meteorol. Soc., 132 (2006), pp. 2483–2504. (Cited on pages 7 and 10.)

[132] ——, *Incremental 4D-Var convergence study*, Tellus A, 59 (2007), pp. 706–718. (Cited on page 21.)

[133] ——, *Model-error estimation in 4D-Var*, Q. J. R. Meteorol. Soc., 133 (2007), pp. 1267–1280. (Cited on pages 7 and 8.)

[134] J. A. Tropp, A. Yurtsever, M. Udell, and V. Cevher, *Practical sketching algorithms for low-rank matrix approximation*, SIAM J. Matrix Anal. Appl., 38 (2017), pp. 1454–1485. (Cited on page 100.)

[135] J. van den Eshof and G. L. G. Sleijpen, *Inexact Krylov subspace methods for linear systems*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 125–153. (Cited on page 41.)

[136] F. Veersé and J.-N. Thépaut, *Multiple-truncation incremental approach for four-dimensional variational data assimilation*, Q. J. R. Meteorol. Soc., 124 (1998), pp. 1889–1908. (Cited on page 20.)

[137] M. Verlaan and A. W. Heemink, *Tidal flow forecasting using reduced rank square root filters*, Stoch. Hydrol. Hydraul., 11 (1997), pp. 349–368. (Cited on page 14.)

[138] J. Verron, L. Gourdeau, D. T. Pham, R. Murtugudde, and A. J. Busalacchi, *An extended Kalman filter to assimilate satellite altimeter data into a nonlinear numerical model of the tropical Pacific Ocean: Method and validation*, J. Geophys. Res., 104 (1999), pp. 5441–5458. (Cited on pages 1 and 15.)

[139] J. A. Verstegen, D. Karssenberg, F. van der Hilst, and A. P. Faaij, *Detecting systemic change in a land use system by Bayesian data assimilation*, Environ. Model. Softw., 75 (2016), pp. 424–438. (Cited on page 1.)

[140] A. J. Wathen, *Preconditioning*, Acta Numerica, 24 (2015), pp. 329–376. (Cited on page 58.)

[141] J. S. Whitaker and T. M. Hamill, *Ensemble data assimilation without perturbed observations*, Mon. Weather Rev., 130 (2002), pp. 1913–1924. (Cited on page 18.)

[142] D. B. Work, O.-P. Tossavainen, S. Blandin, A. M. Bayen, T. Iwuchukwu, and K. Tracton, *An ensemble Kalman filtering approach to highway traffic estimation using GPS enabled mobile devices*, in 2008 47th IEEE Conference on Decision and Control, IEEE, 2008. (Cited on page 1.)

[143] K. Zhou, G. Salomon, and E. Wu, *Balanced realization and model reduction for unstable systems*, Internat. J. Robust Nonlinear Control, 9 (1999), pp. 183–198. (Cited on page 108.)

[144] A. Zilouchian, *Balanced structures and model reduction of unstable systems*, in IEEE Proceedings of the SOUTHEASTCON '91, IEEE, 1991. (Cited on page 108.)

[145] D. Zupanski, *A general weak constraint applicable to operational 4DVAR data assimilation systems*, Mon. Weather Rev., 125 (1997), pp. 2274–2292. (Cited on page 7.)